Z MIKROKOMPUTEREM NA TY

II 7-8

Bajtek

NR INDEKSU 353965
PL ISSN 0860-1674

MAGAZYN KOMPUTEROWY

NR 2 (53–54)'90 CENA 5000



ATARI: 80 ZNAKÓW W WIERSZU

5-cio calowy napęd do FDD 3000

KOMPUTEROWA ENCYKLOPEDIA

ICD DLA ATARI

ORTO-test!

Jak kupować komputery?

JEST DOBRZE, BĘDZIE LEPIEJ

Wiem, nie tylko z własnego doświadczenia, że czytanie tzw. „wstępniaków" nie jest najbardziej pasjonującym doświadczeniem dla Czytelników pisma. Zwłaszcza, jeśli nie są one ciekawe lub składane mało atrakcyjną czcionką. Chcę, aby w przypadku Bajtka było inaczej. Jest naszym wspólnym zamiarem redakcyjnym, aby była to integralna część pisma stanowiącą nie tylko opis zawartości numeru i poczynań zespołu, ale także krótki przegląd aktualnej sytuacji w kraju i na świecie w dziedzinie powszechnie rozumianej informatyki i zagadnień z nią związanych.

To, co mają Państwo przed sobą, to już drugi numer Bajtka, wydany przez nas samodzielnie. Jak na razie nie otrzymaliśmy od Komisji Likwidacyjnej stałego prawa do tytułu, ale dotychczas zawarte porozumienie pozwoli nam towarzyszyć naszym Czytelnikom przynajmniej do końca tego roku. Mamy nadzieję, że dłużej, ale zależy to od ostatecznej decyzji wspomnianej Komisji.

Przeglądając ten numer Bajtka można stwierdzić, że generalna formuła pisma nie uległa zmianie. Zwiększenie objętości do 40 stron umożliwiło wprowadzenie nowego klanu IBM.

W przypadku pozostałych klanów sytuacja wygląda równie dobrze przy czym daje się zauważyć istotny wzrost aktywności klanu Spectrum. Częściowo wiąże się to z drugą młodością, jaką przeżywa ten komputer dzięki coraz popularniejszym systemom dyskowym. Możliwość korzystania z CP/M'u i Turbo Pascala przybliża posiadaczom Spectrum świat profesjonalnych komputerów IBM.

Staramy się też nie stronić od „drutów", czyli opisu drobnych przeróbek sprzętowych. Sądzimy, że takich materiałów nie było w Bajtku za dużo. W tym numerze Bajtka na uwagę zasługuje artykuł poświęcony podłączeniu napędu 5.25" do stacji Timex FDD 3000.

Prowadzenie pisma „klanowego" nakłada pewne obowiązki zarówno na redakcję, jak i na Czytelników. Zdarza się często, że materiały dotyczące jednego klanu mogą być również ciekawe dla posiadaczy innych komputerów. Uwaga ta dotyczy m.in. informacji zawartych w klanie Amstrad na temat odczytu przez te komputery z ZX Spectrum. Namawiamy więc do przeglądania całego numeru. Wprowadzony ponownie spis treści ma ułatwić to zadanie.

Szczególną wagę przywiązujemy do rzetelnego i ciekawego redagowania MicroMagazynu, w którym chcemy dostarczać najnowszych informacji o tym, co dzieje się na świecie. Na razie opieramy się głównie na danych z zachodnich czasopism komputerowych, liczymy jednak w niedalekiej przyszłości na bezpośrednie informacje od producentów.

Czas płynie nieubłaganie, a postęp techniczny, zwłaszcza w dziedzinie komputerów, też domaga się swoich praw. Osobom mającym zamiar dołączyć do klubu komputerów klasy PC dedykuję artykuł z klanu IBM poświęcony zakupom (taniego) sprzętu.

Nie chcemy tracić kontaktu z naszymi Czytelnikami. Jedną z form łączności jest wymiana listów. Dostajemy ich dużo i nie na wszystkie mieliśmy szansę dotychczas odpowiadać. Mamy nadzieję, że w nowych warunkach sytuacja ta ulegnie poprawie. Prosimy jednak, aby osoby oczekujące listownej odpowiedzi na swoje problemy, dołączały do swoich listów zaadresowane koperty zwrotne ze znaczkiem.

A co na świecie? Coraz to szybszy i lepszy sprzęt, unifikacja pewnych idei i technik, dynamiczny rozwój oprogramowania. „Gorące" tematy to multimedia, języki obiektowe i CASE — wspomagana komputerowo inżynieria oprogramowania. Postaramy się w kolejnych numerach Bajtka w sposób przystępny przybliżyć te zagadnienia naszym Czytelnikom.

Z kolei w kraju z pewną ulgą witamy działania Ministerstwa Edukacji Narodowej mające na celu komputeryzację szkolnictwa w oparciu o sprzęt klasy IBM PC. Mamy nadzieję, że widmo nieudanych, spóźnionych i zawodnych, rodem z Elvro, Juniorów przestanie wreszcie spędzać sen z powiek zarówno nauczycieli, jak i uczniów.

Jarosław Młodzki

Bajtek
MAGAZYN KOMPUTEROWY

Redakcja:
ul. Wspólna 61
00-687 Warszawa
Tel. 21-12-05

Redaguje Kolegium:
Marcin Borkowski, Marek Czarkowski, Łukasz Czekajewski, Janusz Jarmoch, Jarosław Młodzki, Waldemar Nowak, Maciej Pietraś, Marcin Przasnyski, Wanda Roszkowska, Wojciech Zientara

Szefowie Klanów:
Amstrad — Jonasz Mayer
Atari — Wojciech Zientara
Commodore — Klaudiusz Dybowski
Co Jest Grane — Marcin Przasnyski
IBM — Marcin Borkowski
Micro Magazyn — Janusz Jarmoch
Spectrum — Maciej Pietraś

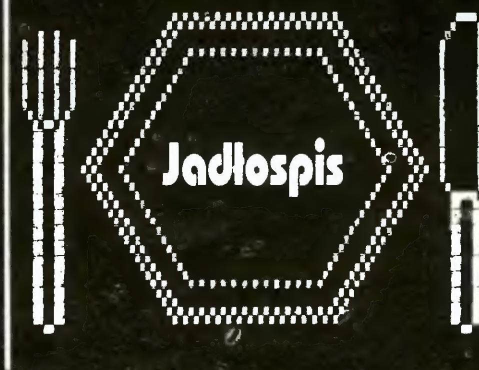
Stali współpracownicy:
Marcin Bójko
Jarosław Burczyński
Piotr Kos
Robert Magdziak
Grzegorz Ostapiuk
Andrzej Pilaszek
Mieczysław Płacheta
Marek Sawicki
Piotr Sumara
Tomasz Tarczyński
Stanisław Winiecki

Opracowanie graficzne:
Wanda Roszkowska

Redaktor techniczny: Maria Radzimińska
Zdjęcia: Leopold Dzikowski

Wydawca:
Spółdzielnia „Bajtek"
ul. Wspólna 61
00-687 Warszawa

Skład i Druk:
Prasowe Zakłady Graficzne w Ciechanowie
Fotoskład: Grażyna Kurzątkowska
Montaż: Grażyna Ostaszewska
Korekta: Maria Krajewska
Teresa Rutkowska
Nakład: 105 tys. egz. Zam. 49610



PIĘKNA I BESTIA -przeczytaj mamie

Komputery przez wiele lat były domeną mężczyzn, jednak w ostatnich dziesięciu latach zauważono, że przy ich obsłudze pracuje coraz więcej kobiet.

W Stanach Zjednoczonych i krajach Europy Zachodniej personel biurowy to głównie młode damy, które zmuszone są do korzystania w pracy nie tylko z komputerów, ale i telefaksów, kserokopiarek, automatów do liczenia pieniędzy, nowoczesnych centralek telefonicznych i innych urządzeń niezbędnych w prowadzeniu interesów. Wysoko wykwalifikowana sekretarka jest poszukiwanym pracownikiem. O jej pozycji decyduje nie tylko znajomość języków obcych, lecz także stopień „oswojenia" z nowoczesną techniką.

Największym problemem w przeszłości było przekonanie personelu o korzyściach płynących z zastosowań komputerów — wspomina Bob Dryer, jeden z dyrektorów pewnej firmy zajmującej się handlem — „Moja sekretarka za nic nie chciała nauczyć się obsługi komputera twierdząc, że zwykła maszyna do pisania jej wystarczy. Dziś jest odwrotnie, panna Smith nie wyobraża sobie pracy bez swojego IBM. Dzięki skomputeryzowaniu przedsiębiorstwa zaoszczędziliśmy tysiące dolarów i poprawiliśmy jakość naszej pracy".



Przykład panny Smith świadczy o istnieniu innego jeszcze zjawiska — wzrostu roli tzw. personelu pomocniczego w przedsiębiorstwie. Dzięki umiejętności posługiwania się sprzętem, wielu pracowników ma dostęp do poufnych często informacji, a także możliwość podejmowania decyzji bez porozumiewania się ze swoimi zwierzchnikami. I robi to!

Znane są przykłady sekretarek samodzielnie kierujących interesami firmy pod nieobecność bossa, dzięki dostępowi do jego komputera. Taka przygoda czasem prowadziła do awansu, ale bywały też straty.

Uczeni amerykańscy zadali sobie pytanie — jakie konsekwencje dla kobiet ma konieczność pracy w środowisku nasyconym nowoczesną techniką? Odpowiedź przeszła ich najśmielsze oczekiwania. Kilkuletnie badania udowodniły, że kobiety bardzo poważnie traktują nowo zdobytą specjalność, są zdyscyplinowane i solidne. Umieją lepiej dbać o stan techniczny urządzeń. Podniósł się ich poziom przygotowania zawodowego.

Jednocześnie dostrzeżono negatywne zjawiska. Nie wskazane jest długotrwałe przesiadywanie przed monitorem. Absolutnie zabronione jest korzystanie z komputerów przez kobiety w ciąży. Promieniowanie elektromagnetyczne ma negatywny wpływ na ich organizm. Niektóre mniej odporne jednostki powinny korzystać z tych urządzeń w sposób umiarkowany. Bywa, że komputer wpływa na psychikę. Niemożność pełnego wykorzystania jego możliwości, albo popełnione błędy potrafią wywołać stres. Nie dzieje się to zbyt często.

Związki komputerów z kobietami zostały zauważone przez producentów filmowych i dwa lata temu w Hollywood powstał film opowiadający dzieje uczucia maszyny do pięknej właścicielki. Jest to oczywiście fikcja, ale obraz się podobał publiczności zmęczonej nieco eksploatowaniem tematu genialnego, trochę szalonego programisty, lub super zdolnego nastolatka.

Kobiety pracują dziś w firmach zajmujących się produkcją oprogramowania. Wymyślają scenariusze gier komputerowych — firma Sierra On-Line, piszą oprogramowanie użytkowe — firma Borland, czy też pracują jako specjalistki od marketingu — rozmiary artykułu nie pozwalają niestety na wymienienie wszystkich przedsiębiorstw zatrudniających damy w tym charakterze.

To prawdziwa rewolucja, jaka zaszła na naszych oczach, stwierdzają autorzy raportu.

W naszym kraju możemy dostrzec początki tych zjawisk. W wielu firmach coraz częściej stosuje się oprogramowanie mające usprawnić pracę księgowości. Działy te są mocno sfeminizowane a co za tym idzie w przyszłości ci sami pracownicy będą zmuszeni do korzystania z komputerów. Mam nadzieję, że jakość ich pracy ulegnie poprawie.

Nie da się tego powiedzieć na przykład o Polskich Kolejach Państwowych — skomputeryzowane kasy na dużych dworcach straszą gigantycznymi kolejkami. Podobnie nie wiadomo po co postawiono w okienkach kasowych Poczty Polskiej komputery klasy IBM PC, skoro wszystkie operacje i tak wykonuje się bez ich udziału.

Zupełnie kuriozalny przypadek to banki, gdzie jakość obsługi klientów bywa różna. Kilka miesięcy temu prasa doniosła, że jeden z warszawskich oddziałów pewnego banku został unieruchomiony, ponieważ szczury przegryzły kable łączące ze sobą komputery w sieć.

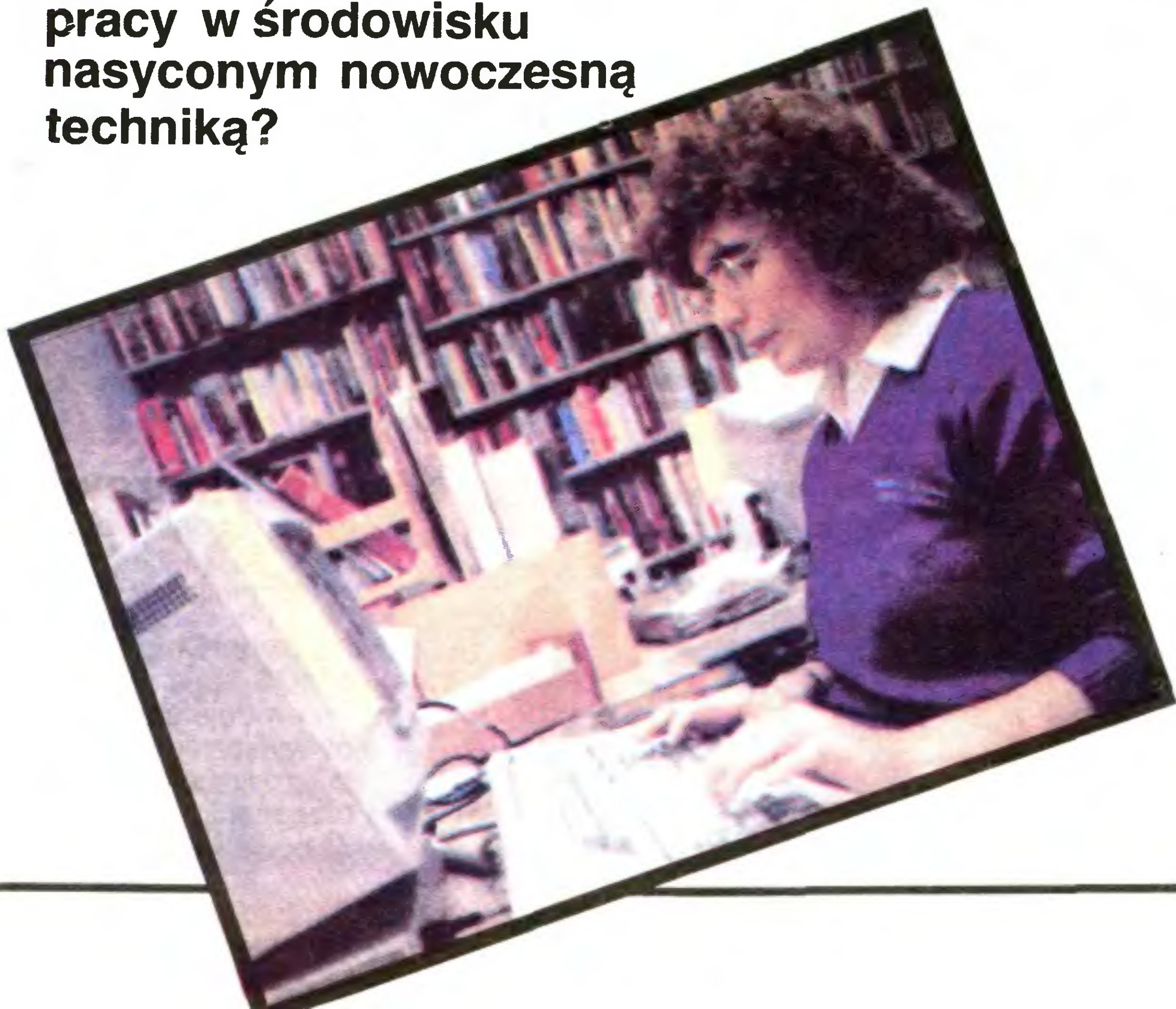
Nie ma to większego związku z kobietami, chyba, że padają one ofiarą rozjuszonych klientów — dworzec PKP, kasy pocztowe, okienka w banku — pragnących być obsłużonymi szybko i sprawnie.

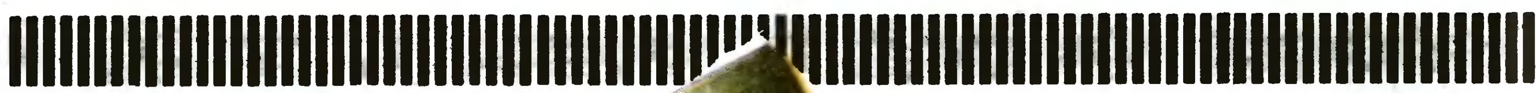
Znacznie lepiej przedstawia się sytuacja w biurach. Jest spokojniej i co najważniejsze jest czas aby zdobyć konieczną wiedzę potrzebną do pracy. O ile „komputeryzacja" przedsiębiorstwa została przeprowadzona sprawnie można przyjąć, że sekretarka, księgowa i kadrowa bardzo szybko docenią zalety nowego narzędzia.

Na zakończenie uwaga do panów. Nie traktujmy z lekceważeniem kobiet. Co prawda nasz system oświaty nie sprzyja jeszcze kształceniu dziewcząt w kierunku informatyki a nauki techniczne są domeną mężczyzn, to jednak za kilka lat zmiany staną się widoczne. Ta część młodego pokolenia, którym rodzice kupili komputery jest inna. Za dziesięć lat, o czym jestem przekonany, będziemy mieli w Polsce bardzo wiele kobiet zajmujących się zawodowo informatyką, o wiedzy i kwalifikacjach znacznie przewyższających niektórych przedstawicieli rodu męskiego. Osobiście jestem ciekaw ile czytelniczek „Bajtka" ma w domu sprzęt? Może napiszecie o tym.

Marek Czarkowski

Jakie konsekwencje dla kobiet ma konieczność pracy w środowisku nasyconym nowoczesną techniką?







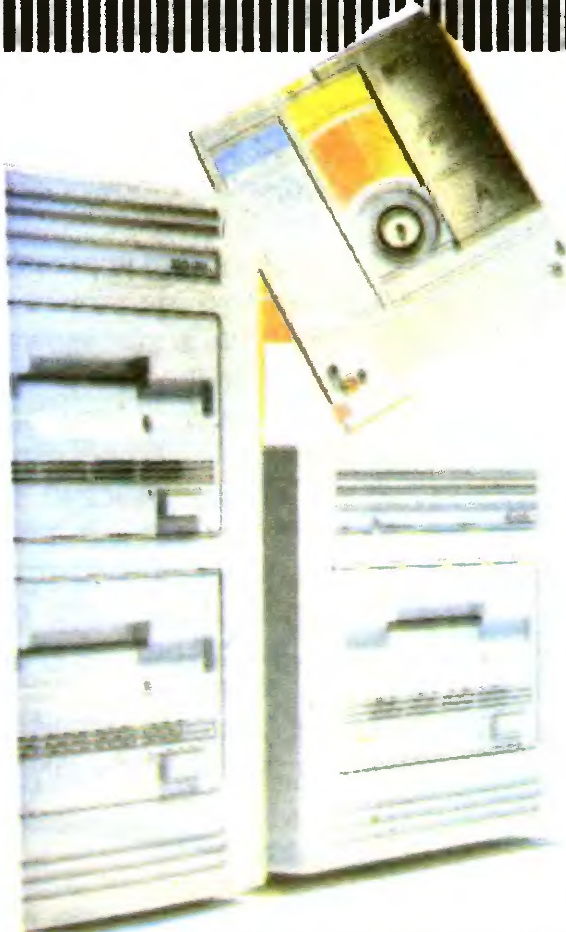
Najcięższy laptop

Toshiba T5200 to prawdziwy czołg wśród laptopów, a jednocześnie przykład wzorowej pracy konstruktorów. W skrzynce o wadze 8,6 kg upakowano mikroprocesor **Intel 80386** taktowany zegarem o częstotliwości 20 MHz i wspierany koprocesorem arytmetycznym **80387** tej samej firmy. Ciekłokrystaliczny ekran o rozdzielczości 640 x 480 pracuje w trybie VGA, EGA, CGA, MDA.

Standardowe 2 MB RAM można rozszerzyć do 8 MB, a twardy dysk 40 MB można zastąpić innym o pojemności 100 MB. Całości dopełnia pojedyńczy napęd dysków elastycznych 3,5 cala, standardowe złącza **Centronics, RS-232C, RGB.** Zasilanie gwarantuje ładowany z sieci akumulator. Klawiatura typu „QWERTY" liczy 92 klawisze. Komputer korzysta z systemu operacyjnego **MS DOS 4.01,** ale przewidziano możliwość zainstalowania systemu **MS OS/2 1.1.**

Firma liczy, że ten najpotężniejszy z rodziny laptopów Toshiby zyska uznanie bogatych biznesmenów i naukowców.

(M C)



DYSK OPTYCZNY firmy PINNACLE MICRO

W Polsce magnetyczny dysk twardy o pojemności 80 MB jest często obiektem marzeń nie tylko indywidualnych użytkowników komputerów, ale także i wielu instytucji. Od momentu wprowadzenia na rynek bardzo innowacyjnego produktu jakim był komputer NeXT pana Jobsa, szlagierem pamięci masowych stały się dyski optyczne pozwalające, w odróżnieniu od poprzednich konstrukcji, na wielokrotny zapis i kasowanie w sposób optyczny informacji.

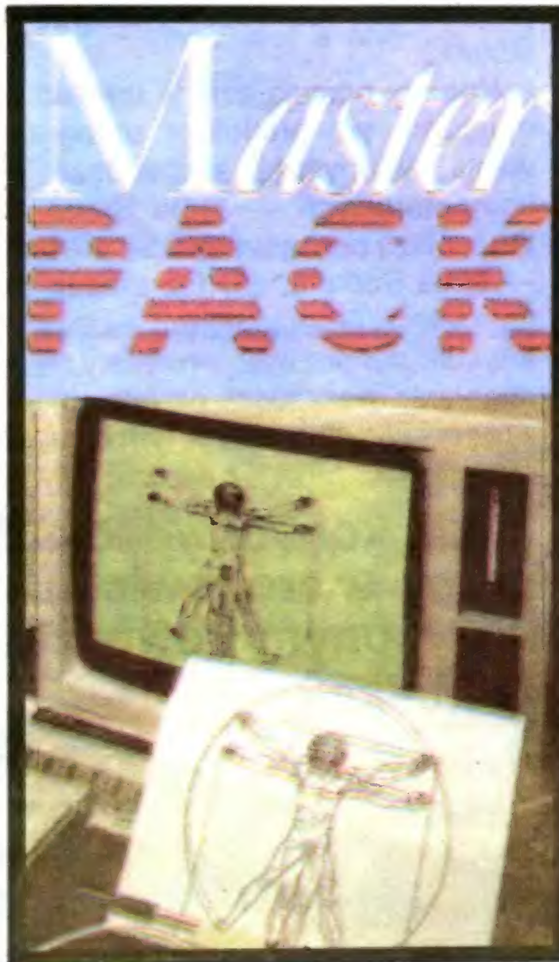
Typowym produktem w tej klasie jest REO-650 — optyczny dysk twardy firmy Pinnacle Micro. Wymienne dyskietki (cartridge) — przypominające płyty kompaktowe — pozwalają na zapis 650 MB danych. Czasy dostępu zbliżone są do tych, z jakimi mamy do czynienia w przypakdu wolnych dysków magnetycznych, ale ilość przechowywanych informacji, wymienność i bezpieczeństwo danych powodują, że urządzenia te stają się coraz bardziej popularne. Jak zwykle pewną przeszkodą jest wysoka cena nowości — 5000 USD.

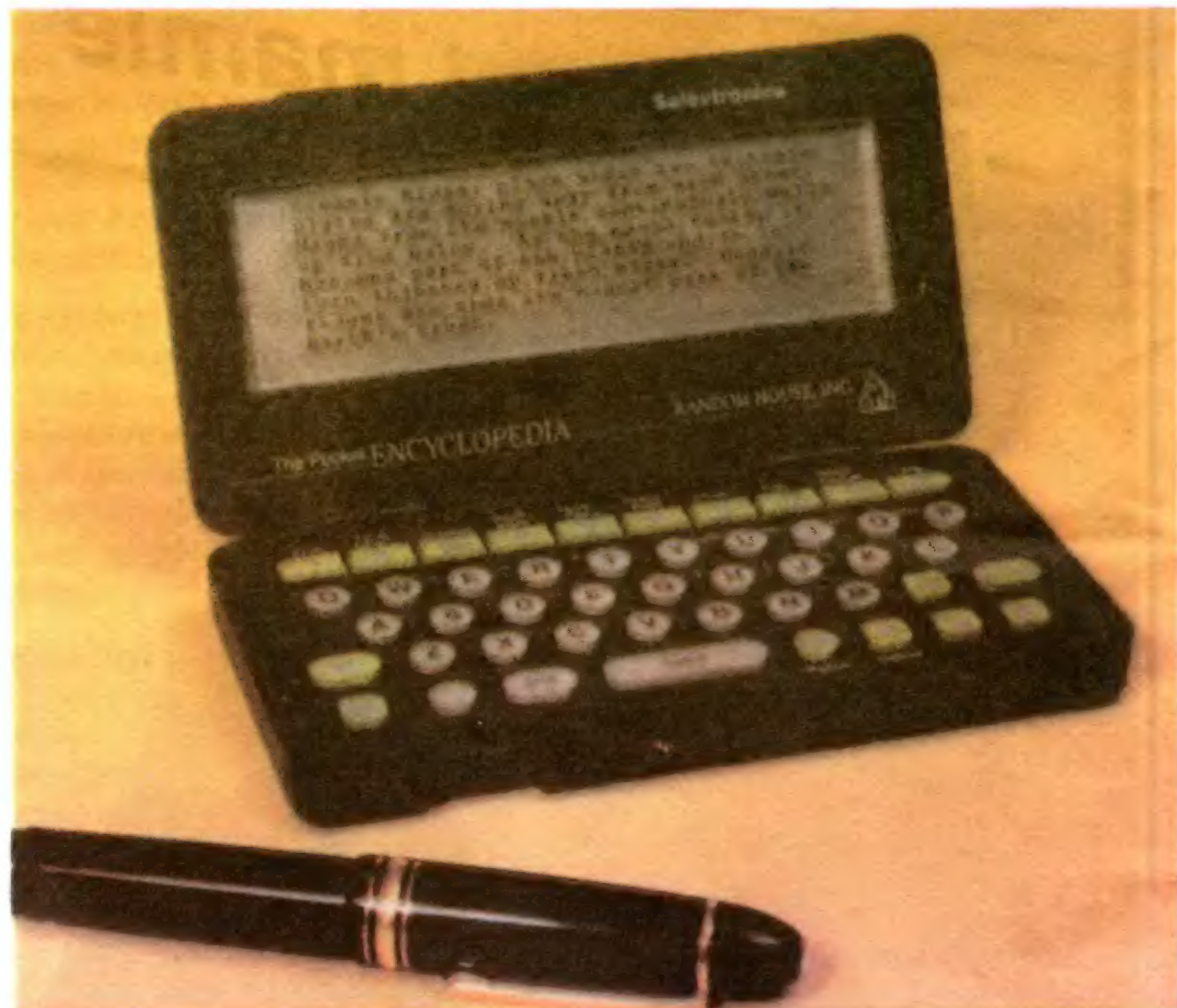
(im)

SKANER DO PCW

Jedną z zalet komputerów serii Amstrad PCW jest oferowana w cenie zestawu drukarka. Dzięki skanerowi proponowanemu przez angielską firmę Database Software, możliwe jest dalsze zwiększenie jej możliwości. Urządzenie to ułatwia wprowadzanie do komputera rysunków, schematów, fotografii i map. Oprócz samej głowicy skanującej, mocowanej do głowicy drukującej, oferowane są także dwa programy wspomagające — Master Scan i Master Paint. Cały zestaw, pod nazwą Master Pack, sprzedawany za 57 funtów, stanowi względnie tanią alternatywę w stosunku do podobnych systemów dostępnych na komputerach IBM PC.

(JM)





ELEKTRONICZNA ENCYKLOPEDIA

Pierwszą w świecie kieszonkową elektroniczną encyklopedię wprowadziła na rynek amerykańska firma **SelecTronics** z Minneapolis. Miniaturowy komputer zaopatrzony jest w ekran ciekłokrystaliczny wyświetlający po 40 znaków w ośmiu liniach. Klawiatura typu „Qwerty" liczy 47 klawiszy. Pamięć zawiera około 8 megabajtów informacji! Właściciel korzysta ze „zwykłej" encyklopedii, notesu, zegara z budzikiem i kalkulatora. Przewidziano możliwość korzystania z modułów zawierających inne rozszerzenia.

Elektroniczna encyklopedia opracowana została we współpracy ze znanym wydawnictwem **Random House, INC.** i bazuje na wydawanej przez tę firmę **The Random House Encyclopedia**, sprzedanej w ponad 200 000 egzemplarzy.

Firma produkuje także **Elektroniczną Biblię** i **Słownik** zawierający ponad 100 000 haseł, reklamując swoje produkty jako idealny prezent dla studentów i biznesmenów.

(MC)

Super Chip

Inżynierowie z **TRW i Motoroli** pracujący na zlecenie Departamentu Obrony USA oświadczyli, że sukcesem zakończyły się ich prace nad nowym mikroprocesorem.

„SuperChip" wykonany został w technologii **CMOS** i ma moc obliczeniową rzędu **200 MIPS.** Przewiduje się, że początkowo będzie produkowany wyłącznie na potrzeby Armii. Jeśli trafi „do cywila" znajdzie zastosowanie głównie w komputerach przeznaczonych do diagnostyki medycznej, wspomagania projektowania CAD i przetwarzania obrazów.

Eksperci spodziewają się, że wkrótce największy konkurent **Motoroli** firma **Intel** przedstawi własny produkt **Intel 80786 CPU** o którym jak dotychczas „wiele się mówiło", ale nikt go nie widział.

Przykłady te skłaniają do wniosku, że w niedalekiej przyszłości będziemy dysponowali komputerami o rozmiarach IBM PC i mocy obliczeniowej superkomputerów **CRAY.**

(MC)



ATARI LYNX

Jack Tramiel zapewne dowiedział się, że wielki wschodni koncern Elektronika produkuje „kieszonkowe" gry wideo — z Wilkiem i Zającem w rolach głównych — masowo kupowane przez Polaków i tak się tym przejął, że „pogonił" swoich inżynierów, aby wyprzedzili Rosjan. Owocem ich starań jest **Lynx.**

Atari pod wodzą **Nolana Bushnella** startowała jako firma produkująca automaty do gier video. **Tramiel** po sukcesach serii **Atari 800, Atari 65, 130** a następnie **Atari ST** „wrócił do źródeł".

Lynx jest typową „maszynką do grania" wyposażoną w kolorowy ciekłokrystaliczny ekran i głośnik. Gry „ładowane" są z cartritge'a, tak więc pirackie kopiowanie jest niemożliwe. Dobra rozdzielczość ekranu sprawia, że nowy pomysł Jacka sprzedaje się nieźle w USA. Trudno dziś orzec czy przyjmie się on w Polsce — jest drogi i mam wrażenie, że my gustujemy w pełnokrwistych komputerach, zabawki pozostawiając wschodniemu koncernowi Elektronika.

(MC)

LAPTOPY FIRMY AMSTRAD

Na początku tego roku firma Amstrad wypuściła na rynek brytyjski dwa nowe modele komputerów typu laptop. Jednym z nich jest ALT-286, wyposażony w procesor AMD 80L286, 1MB RAM, napęd dysku 3.5" o pojemności 1.44 MB i dysk twardy 20MB z czasem dostępu 28 ms. Ciekłokrystaliczny ekran realizuje standard VGA o rozdzielczości 640∗480 przy 32 odcieniach szarości. Zachowano zgodność z innymi trybami graficznymi: CGA, EGA, MDA i Hercules.

Cały komputer łącznie z baterią akumulatorów pozwalających na 2-godzinną pracę waży 7 kg i ma rozmiary 318∗380∗95 (mm). Procesor pracuje z częstotliwością 16 MHz, a pamięć RAM o czasie dostępu 60 ns pozwala na pracę bez cykli oczekiwania (0 wait states) i może być rozszerzona do 4 MB. Możliwe jest dodanie koprocesora i jednej karty rozszerzającej 16-bitowej. Standardowym wyposażeniem są dwa złącza szeregowe (RS 232C) i jedno równoległe (Centronics). Przy pracy stacjonarnej możliwe jest dołączenie monitora kolorowego typu VGA, zwykłej klawiatura AT 101 klawiszy i drugiego napędu dysków elastycznych (3.5" lub 5.25").

Model ALT-386SX różni się od poprzedniego procesorem (Intel 80386SX) i zastosowanym dyskiem twardym (40MB i 25 ms). Oba komputery sprzedawane są z systemem operacyjnym MS-DOS 3.30.

(JM)



VIDEO TELEFON



W roku ubiegłym inżynierowie z firm **Vistacom Industries** i **Nokia Telecommunications** zakończyli trwające od 1986 prace nad opracowaniem videofonu. **VISTACOM ISDN** — taką nazwę nosi urządzenie jest *„eleganckim" ekonomicznym i zaawansowanym technologicznie produktem niezbędnym w kierowaniu firmą i biznesie".*

Videotelefony nie są nowymi urządzeniami, to zwraca uwagę zwartą budową przystosowaną do warunków biurowych. Jest ono przygotowane do pracy w systemie telekonferencji, a także z komputerem.

Wydaje się, że videotelefony mają przed sobą przyszłość. Dziś ich cena jest jeszcze wysoka, ale konkurencja i rozwój technologii spowodują jej spadek. Prezentowany model produkowany jest w Finlandii.

(MC)

DTP i mysz do PCW

Nowością na rynku oprogramowania komputerów Amstrad PCW jest program Micro Design 2 sprzedawany przez firmę Creative Technology. Łącznie z myszą firmy AMX kosztuje on ponad 80 funtów. Za tę cenę otrzymujemy sensowny zestaw do prac typu DTP (Desktop Publishing) o parametrach znacznie przewyższających dotychczasowych konkurentów tzn. programy Stop Press, The Desktop Publisher i Newsdesk International. Oprócz 4-krotnie większej, w stosunku do Stop Press, rozdzielczości otrzymujemy także programy sterujące (ang. driver'y) do drukarek 24-igłowych i laserowych.

(JM)

KOMPUTEROWA ENCYKLOPEDIA

CO TO JEST KOMPUTER?

Komputer jest to urządzenie elektroniczne, którego działanie można programować. Dzięki temu jest bardzo wszechstronny i może robić różne rzeczy. Jakie? To zależy od konfiguracji komputera (a więc od tego, z czego jest zbudowany i jakie ma dołączone urządzenia dodatkowe), oraz od programu, który steruje jego pracą.

JAK KOMPUTER JEST PROGRAMOWANY?

Komputer potrafi rozpoznać i wykonać polecenia napisane w ściśle określony sposób. Takie polecenie, zwykle w postaci słowa lub kilku słów nazywamy **rozkazem**.

Zbiór rozkazów — czyli zbiór słów, które mówią komputerowi co ma zrobić — nazywa się **programem komputerowym**.

Wszystkie rozkazy razem (a jest ich bardzo dużo), to tak zwany **język komputerowy**.

Język komputerowy różni się od języka ludzkiego i trzeba się go nauczyć. Najszybciej można się nauczyć języka komputerowego o nazwie **BASIC**. Jego rozkazy są podobne do normalnych — ludzkich. Niestety w języku BASIC nie można programować po polsku. Rozkazy w nim pochodzą z języka angielskiego (np. SOUND znaczy dźwięk).

Rozkaz może być wykonany od razu — pojedyńczo. Ale gdy napiszemy słowo-rozkaz, to komputer nie wie, czy już skończyliśmy, czy też będziemy jeszcze coś pisać. Dlatego zawsze po napisaniu rozkazu musimy nacisnąć klawisz z napisem **ENTER** (wprowadź). Jest to sygnał dla komputera, że skończyliśmy pisać rozkaz i ma on być teraz wykonany.

Możemy też zapamiętać dużo rozkazów i wykonać je szybko po sobie. Jeśli tak, to przed rozkazem trzeba napisać jego numer. Dzięki temu komputer wie, w jakiej kolejności ma wykonywać rozkazy. (Nie musimy ich wpisywać w kolejności wykonywania). Aby komputer wykonał tak napisany program, musimy na klawiaturze napisać rozkaz **RUN** (biegnij, naprzód).

Rozkazy mogą być różne (np. wydrukuj, oblicz). Oprócz rozkazu musimy podać, co komputer ma wydrukować, obliczyć, itp. A oto przykład rozkazu w języku BASIC:

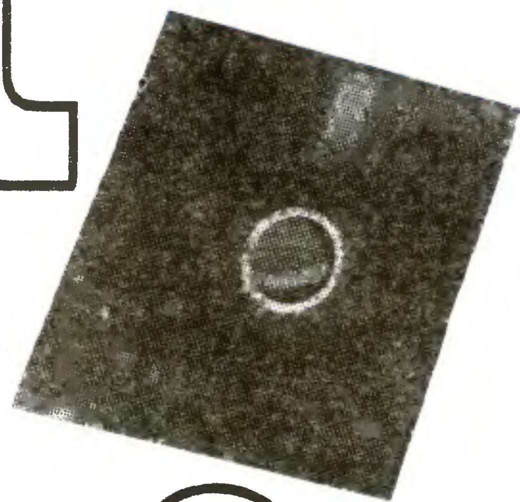
PRINT 4 + 2

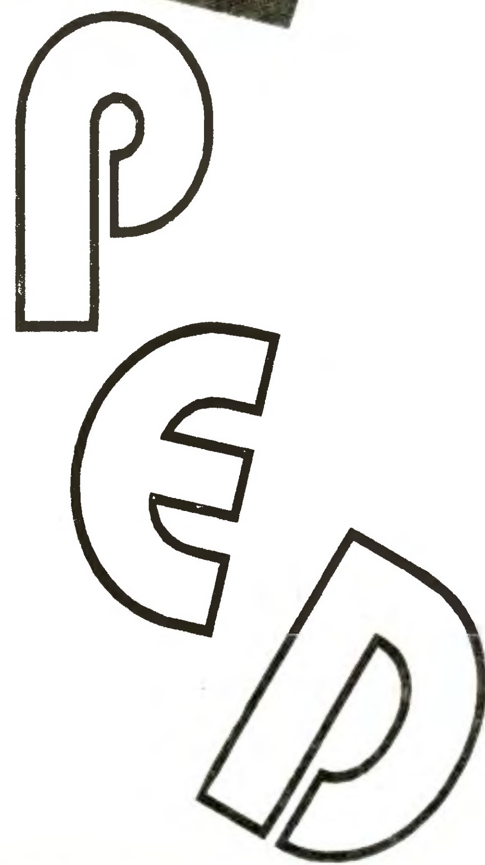
„PRINT" oznacza: wydrukuj, wyświetl na ekranie. Ale co? Rzecz jasna, że wynik działania 4 + 2 (czyli po prostu 6). Jeśli więc napiszemy: PRINT 4 + 2 i wciśniemy klawisz ENTER to na ekranie zobaczymy:

6
READY

Napis READY znaczy gotowe, skończyłem. Komputer informuje nas w ten sposób, że możemy wprowadzać następny rozkaz. Trzeba się nauczyć innych rozkazów i... można programować dalej!







JAK JEST ZBUDOWANY KOMPUTER?

Komputer jest zbudowany z wielu części. Nie wszystkie z nich widać. Wiele elementów jest schowanych w obudowie komputera. Inne widać doskonale. Są to:

Obudowa

Chroni delikatne układy elektroniczne komputera.

Klawiatura

Dzięki niej możemy „mówić" do komputera. Komputer rejestruje każde naciśnięcie klawisza. Dzięki temu możemy wydawać komputerowi rozkazy. W czasie wykonywania programu komputer może nie czytać klawiatury. Musimy wtedy czekać, aż skończy pracę, albo zezwoli na wprowadzanie danych do programu.

Monitor (albo telewizor)

Za jego pomocą komputer „mówi" do nas. Wyświetla na nim różne wiadomości lub obrazy. Ich treść zależy od tego, jak jest zaprogramowany komputer. Informacjom na ekranie towarzyszą często dźwięki z głośniczka. Mają one zwrócić nam uwagę na to, że na ekranie coś się dzieje.

Stacja dysków (albo magnetofon)

Służy do zapisywania i odczytywania programów. Stacja dysków to prawie magnetofon, trochę innej konstrukcji. **Dysk elastyczny** albo inaczej dyskietka to krążek plastikowy pokryty cienką warstwą magnetyczną. Stacja dysków działa dużo szybciej niż magnetofon.

CO JEST W ŚRODKU KOMPUTERA?

Komputer — nawet ten bardzo niewielki — jest bardzo skomplikowanym urządzeniem. Może być taki mały, ponieważ do jego budowy użyto wielu układów scalonych. W obudowie komputera znajdują się jego najważniejsze elementy. Bez nich nie byłoby komputera.

Układy scalone

To czarne plastikowe kostki (wielkości małej gumki do wycierania), z metalowymi nóżkami. Poprzez te nóżki płyną sygnały elektryczne do elementów elektronicznych (tranzystorów) zamkniętych w środku kostki. A może ich tam być nawet 100.000 i więcej! Układy scalone z zewnątrz wyglądają prawie jednakowo. Ale różnią się tym, co jest w środku, i służą do różnych rzeczy.

Zasilacz

Dostarcza prądu dla całego komputera. Od jego włączenia zaczynamy zawsze nasze spotkanie z komputerem. Po pracy trzeba pamiętać, aby go wyłączyć.

Procesor

To specjalny układ scalony. Największy i najbardziej skomplikowany ze wszystkich. Procesor jest mózgiem komputera. Kieruje jego pracą i wykonuje program — pobiera rozkazy od operatora i wykonuje je, kieruje dane na ekran, czyta i zapisuje pamięć.

Pamięć

To miejsce, gdzie przechowuje się informacje dla procesora. To jakby biblioteka z książkami i zeszytami. Jeśli procesor potrzebuje informacji, to sięga do pamięci, tak jak Ty sięgasz po książkę. (Mówimy wtedy, że **procesor czyta pamięć**). Jeśli wynikiem działania procesora jest nowa informacja (np. wynik obliczeń), to komputer zapisuje ją do pamięci, tak jak Ty pracę domową do zeszytu. (Mówimy wtedy, że **procesor zapisuje pamięć**).

Głośniczek

Oczywiście może wygrywać melodyjki. Ale najważniejsze jest to, że daje sygnał PIIP!!, gdy robimy coś nie tak (Np. gdy naciskamy zły klawisz).

Inne części

Zasilacz, procesor i pamięć musi mieć każdy komputer. Inne części tylko wspomagają ich pracę i nie są takie ważne. Dlatego nie będziemy o nich mówić.

Prawie każdy z nas zetknął się już z komputerem. Wiemy, ile czasu można spędzić przy nim na komputerowych grach. Ale co to jest komputer, do czego służy (oprócz gier), z czego jest zbudowany i jak działa? Jeśli chcesz znać odpowiedzi na te pytania, to pomoże Ci w tym ta mała encyklopedia.



JAK DUŻA JEST PAMIĘĆ W KOMPUTERZE?

Pamięć komputera nie ma stron i liter. Ma tylko układy scalone i sygnały elektryczne. Ale te sygnały procesor umie zamienić na litery wyświetlane na ekranie. Dlatego wielkość pamięci określa się w bajtach. Jeden BAJT to pamięć, w której można zapisać 1 literę.

1 **BAJT** = 1 litera

Komputer może zapamiętać dużo więcej, dlatego częściej używa się kilobajtów (w skrócie **KB**).

1 **KILOBAJT** = 1024 * BAJT = 1024 litery

A więc w jednym kilobajcie pamięci można zapisać 1024 litery.

Np. komputer AMSTRAD CPC 6128 ma 128 KB pamięci. Można więc w nim zapisać program złożony z 128 * 1024 = 131.072 liter! (Jest to około 150 stron maszynopisu).

Rozumiesz teraz, dlaczego program ładowany z kasety magnetofonowej — literka po literce — ładuje się tak długo? Komputer, który ma większą pamięć, może wykonywać dłuższe i bardziej skomplikowane programy. Jest więc lepszy (ale i droższy!) od tego z mniejszą pamięcią.

JAKIE RODZAJE PAMIĘCI MA KOMPUTER?

Pamięć ROM

ROM jest pamięcią tylko do czytania. Procesor nie może nic do niej zapisać, ale za to jej zawartość nie jest tracona po wyłączeniu zasilania. Zawiera ona program obsługi komputera wpisany do niej raz na zawsze w fabryce. To dzięki tej pamięci i zapisanemu w niej programowi komputer po włączeniu reaguje na klawisze, pisze na ekranie i umie załadować programy z grami.

Pamięć RAM

RAM jest pamięcią, którą procesor może czytać, zapisywać, kasować i znowu zapisywać. Tam wpisywane są programy, które ładujesz z kasety magnetofonowej lub z dyskietki. Tam też jest wpisywany program, który pisze programista. Ta wspaniała pamięć niestety gubi (zapomina) swoją zawartość po wyłączeniu zasilania. Dlatego przed wyłączeniem zasilania musimy napisany program przepisać na taśmę magnetofonową lub dyskietkę. (Mówimy **nagrać program**.) A po włączeniu komputera musimy ten program znów wpisać (**załadować**) do pamięci. Niewygodne? Może i tak. Ale dzięki temu może to być za każdym razem inny program!

JAKIE URZĄDZENIA MOŻNA DOŁĄCZYĆ DO KOMPUTERA?

Drukarka

Drukuje literki i obrazki za pomocą malutkich kropeczek. Drukowanie odbywa się bardzo szybko. Ten sam tekst lub obrazek można drukować wiele razy. Niektóre drukarki mogą drukować kolorowe rysunki. Drukarki są bardzo drogie i kosztują niewiele mniej od komputera!

Ploter

Rysuje rysunki za pomocą pisaka podobnego do flamastra. Pisak jest przesuwany po prowadnicach za pomocą silniczków elektrycznych. Najczęściej jest kilka różnokolorowych pisaków. Są one automatycznie zmieniane, przesuwane, opuszczane na papier i podnoszone. Może on także malować piękne rysunki. Ale niestety! Ploter kosztuje więcej niż komputer i kupują go tylko bogaci profesjonaliści. Używa się go do wykonywania dokładnych rysunków maszyn i urządzeń.

Joystick

Działa jak drążek sterowy w samolocie. Najczęściej używany w czasie gier do poruszania się po ekranie komputera. Są też programy, które pozwalają za jego pomocą np. malować obrazki.

Modem

Pozwala podłączyć komputer do telefonu. Jeśli zatelefonujemy do drugiego komputera wyposażonego w modem, to komputery mogą ze sobą „rozmawiać". Można w ten sposób przesyłać programy, teksty, obrazki. Komputery robią to bardzo szybko i taka „rozmowa" prawie nic nie kosztuje. Specjalny program umożliwia komputerowi nadawanie i odbieranie wiadomości nawet wtedy, gdy nas nie ma w domu, albo gdy śpimy. Musimy tylko zaprogramować, gdzie i kiedy ma telefonować! Odebrane przez komputer informacje są zapamiętywane i możemy je obejrzeć wtedy, gdy mamy czas.

To nie wszystko

Czy słyszeliście o myszy? Albo o piórze świetlnym? Ale na razie wystarczy. I tak było tego bardzo dużo.

DO CZEGO SŁUŻY KOMPUTER?

To bardzo trudne pytanie. Bo komputer może robić wszystko. Potrzebuje tylko różnych programów i narzędzi. Jeśli zrobimy komputerowo sterowany młotek, to komputerem można nawet wbijać gwoździe. Tylko po co? Spróbujmy więc pokazać do czego komputery są używane najczęściej.

Obliczenia matematyczne

Od tego się zaczęło! Już pierwsze komputery liczyły dużo szybciej i dokładniej niż człowiek. Pozwoliły zaprojektować nowoczesne samoloty i rakiety. Obliczyły tor lotu człowieka na Księżyc. I liczą nadal — dla inżynierów, fizyków i matematyków. Liczą w fabrykach, w laboratoriach, w bankach. Liczą coraz więcej i szybciej. Brak pracy im nie grozi!

Sterowanie i pomiary

Sterują światłami na skrzyżowaniach i całymi fabrykami. Robotami przy produkcji samochodów i sztucznym sercem w czasie operacji. Wyposażone w odpowiednie czujniki zarejestrują co tylko chcemy: trzęsienie ziemi i tupnięcie komara. Wyniki pomiarów mogą być albo zapamiętane, albo od razu wykorzystane (np. do sterowania). Wszystkie te czynności komputer wykona bardzo szybko i bezbłędnie!

Pomoc w projektowaniu

Pomagają kreślić rysunki i określać wytrzymałość elementów. Projektują schematy płytek z elementami elektronicznymi i konstruują budynki. Ołówek i linijkę zastąpił ekran komputera i drukarka. Na komputerze nie robią się kleksy, a zmian w projekcie można dokonywać nawet sto razy. Aż otrzymamy to, co chcieliśmy zaprojektować.

Banki danych

Służą do szybkiego wyszukiwania potrzebnej informacji. Chcesz lecieć samolotem lub jechać pociągiem? Komputer sprawdzi, czy są miejsca i wyda Ci bilet. Jeśli zachorowałeś to podaj mu objawy choroby, a znajdzie lekarstwo i określi sposób leczenia. Już nie musisz kupować wielotomowych encyklopedii! Możesz je mieć w komputerze. Jeśli masz modem, to masz dostęp do informacji zapisanej we wszystkich dużych komputerach całego świata!

Edytory tekstów

To inteligentne maszyny do pisania. Potrafią poprawić błąd ortograficzny. Mogą szybko zmienić układ tekstu i zrobić miejsce na nowe rysunki. Znajdą wskazany wyraz i zamienią innym. Przepiszą zaproszenie lub list tyle razy, ile trzeba. Dzięki nim każdy może mieć swoją gazetę, a prawdziwe gazety są wydawane szybciej i ładniej.

Nauczanie

Komputer poprowadzi lekcję i sprawdzi, czy ją zapamiętałeś. Jest cierpliwy i ma zawsze czas. Może egzaminować i stawiać oceny. A gdy postawi dwóję, to... można go wyłączyć!

Do prowadzenia prawdziwej lekcji komputerowej potrzeba wielu komputerów. Najlepiej, żeby każdy uczeń w klasie miał przed sobą osobny komputer. Komputery uczniowskie są połączone z komputerem nauczyciela. Dzięki temu nauczyciel może ładować programy jednocześnie do wszystkich komputerów uczniowskich. A potem może na swoim ekranie oglądać po kolei co robi każdy z uczniów. Bo każdy z uczniów może robić co innego i nie przeszkadza innym. Dobry uczeń rozwiąże szybko komputerowe zadanie i zostanie mu jeszcze czas na grę-nagrodę. Mniej zdolni uczniowie będą mogli w tym czasie spokojnie dokończyć zadanie. Bo komputer nie zna litości i nie da się go oszukać!

Komputery łączy się ze sobą nie tylko w szkole. Robi się tak samo np. w bankach i biurach. Taki sposób połączenia komputerów — że mogą pracować wszystkie razem, albo każdy osobno — nazywamy **siecią komputerową**.

Rozrywka

I wreszcie gry komputerowe. Ale nie tylko „goń i strzelaj". Także programy do rysowania obrazków, komponowania muzyki, gry logiczne (np. szachy lub brydż) i wiele, wiele innych.

CZY TO WSZYSTKO?

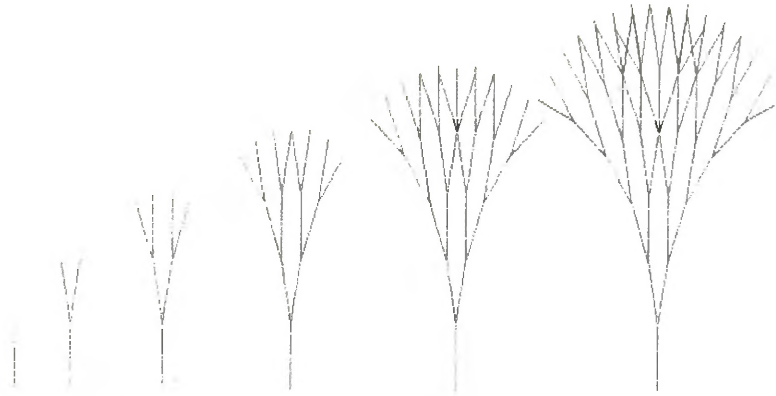
Na pewno nie! Ale jeśli uważnie przeczytałeś **ENCYKLOPEDIĘ KOMPUTEROWĄ** i zapamiętałeś wiadomości w niej zawarte, to możesz przyznać sobie tytuł **KOMPUTEROWEGO EKSPERTA**.

Może zaczniesz się teraz uczyć programowania? Jeśli tak, to polecam Ci język programowania **LOGO**. Jest łatwy i opracowany specjalnie dla dzieci. A co najważniejsze — opracowano polskie LOGO, które czyta, pisze i rozmawia z Tobą po polsku! LOGO ma dla Ciebie bardzo grzecznego żółwia, który bardzo lubi rysować. Radzę się z nim zapoznać.

Powodzenia!

Tadeusz Panasiewicz

Pierwszy znany przepis na drzewo powstał w sposób naturalny około czterystu milionów lat temu. Przez następne 260 milionów lat nie ulegał większym zmianom, aż pod koniec mezozoiku natura zdecydowała się zamknąć go w skorupce. W ten sposób powstały rośliny okrytozalążkowe, a przepis na drzewo stał się twardym orzechem do zgryzienia. Od tamtego czasu w dziedzinie przepisów na drzewo panował do niedawna zastój.



Rys. 1. Kolejne fazy wzrostu drzewa

PRZEPIS na DRZEWO

Przełamali go matematycy, do których po chwili dołączyli informatycy. Ich drzewo opisał w zeszłym roku w grudniowym numerze Bajtka Andrzej Krul. Jego opis — choć fachowy — nie był jednak przepisem, pozwalającym na skonstruowanie drzewa podobnego do czegokolwiek rosnącego w lesie. Dlatego też, o ile postawimy sobie takie zadanie, musimy szukać dalej.

Przyjmijmy, że interesuje nas drzewo zimowe, bez liści. Takie postawienie sprawy daje podwójną korzyść. Po pierwsze, algorytm którym się posłużymy będzie znacznie prostszy, po drugie, do testowania naszego przepisu wystarczy monitor monochromatyczny bursztynowy, którego kolor jest zbliżony do barwy pnia. Jeżeli ktoś z naszych czytelników ma monitor zielony, powinien zająć się przepisem na paproć, który mamy nadzieję wkrótce zamieścić.

Żeby nie trzymać Was w napięciu, od razu przedstawię propozycję definicji drzewa, którą następnie przerobimy na przepis. Otóż uważam, że **drzewo składa się z pnia, z którego wyrastają drzewo w prawo i drzewo w lewo.** Definicja zwarta, krótka, i — choć wielu nie będzie się podobać, zwłaszcza informatykom i botanikom, dopatrującym się zbrodni w pozbawieniu drzewa korzenia — w zupełności wystarczająca, jeśli ktoś chce przy jej pomocy zrobić drzewo. Zanim się tym jednak zajmiemy, porównajmy powyższą definicję z tym, co widać w lesie.

Drzewo składa się z pnia — tu chyba nie ma żadnych wątpliwości; **z którego wyrastają** — to też jest prawda, z każdego pnia wyrastają różne rzeczy w różne strony; **drzewo w prawo i drzewo w lewo** — tu musimy zatrzymać się na dłużej. Już słyszę podstawowy zarzut — jak może wyrastać drzewo z drzewa? Zaraz wam udowodnię że może. Jeżeli ułamiemy gałąź wierzby i wsadzimy w ziemię, to co z niej wyrośnie — drzewo! A gdzie przedtem ta gałąź rosła — na drzewie. Czyli drzewo na drzewie rosnąć może, czyli definicja jest w porządku.

Teraz trzeba ją przerobić na przepis. Przyjrzyjmy się naszemu drzewu (na rysunku 1). Najpierw jest sam pień. Z pnia wyrastają dwa następne — jeden rośnie w lewo, drugi w prawo. Teraz z tego lewego znowu wyrastają dwa — jeden w lewo, drugi w prawo, to samo dzieje się z prawym drzewem. Pień nie musi się o nic martwić — jego jedynym zadaniem jest wypuszczenie dwóch identycznych ze sobą pędów w dwie strony świata.

Potrzebna będzie jednak jedna poprawka. Nasze drzewo nie ma końca, i rośnie bez opamiętania, wypuszczając coraz to nowe pędy. Prawdziwe drzewo wprawdzie też rośnie bez opamiętania, ale prędzej czy później z jakiegoś powodu usycha, i przestaje rosnąć. Żeby zmusić do tego również nasze drzewo, dołóżmy do przepisu ograniczenie — zabrońmy któremuś kolejnemu pniu wypuszczenia pędów. Dzięki temu drzewo urośnie do pewnej, określonej przez nas wysokości.

Zamieńmy teraz nasz przepis na program. Na przykład w Pascalu.

Program główny służy tylko do przygotowania trybu graficznego i zasiania drzewa, które dalej rośnie samo. Zauważcie, że procedura **drzewo**, po narysowaniu pnia, wywołuje w celu narysowania następnych pokoleń pędów samą siebie z nieco zmodyfikowanymi danymi wejściowymi. (Jest to zgodne z definicją, której częścią jest ona sama). Te dane to współrzędne punktu, z którego wyrasta nowy pień, jego kierunek, czyli kąt między pniem a powierzchnią ziemi (wybraną jako płaszczyzna odniesienia) i w końcu informacja o tym, ile jeszcze pokoleń pędów może wyrosnąć.

Taki proces, którego częścią jest on sam, nazywa się rekurencyjnym, i jest często niesłychanie wygodny. Kto nie wierzy, niech spróbuje narysować nasze drzewo w inny sposób. Wprawdzie matematycznie udowodniono, że każdy algorytm da się zrealizować przy użyciu pętli **while** i zmiennych pomocniczych, ale próby tłumaczenia algorytmów rekurencyjnych na takie pętle pozostawić trzeba masochistom. Jest to jeden z wielu powodów, dla których odradzam wszystkim używanie BASICA. Kto nie rozumie dlaczego, niech spróbuje zrealizować w nim nasz algorytm ry-

```
{ Turbo Pascal 4.0 lub późniejszy. Inne kompilatory wymagać
będą pewnych niewielkich przeróbek, głównie związanych z
odwołaniami do procedur graficznych. }

uses hercules, graph, crt;

const
  n_pokol   = 10;          { Ilość pędów, które mogą wyrosnąć. }
  krok      = 15;                              { Długość pędu. }
  pkmp      = 0.3;                { Połowa kąta między pędami. }

var
  crdmd : integer;

procedure drzewo(x, y, kierunek : real;dalej :word);
var
  nowex, nowey : real;
begin
  dalej:=dalej-1;
  if dalej=0 then EXIT;                { Więcej pędów nie będzie. }
  nowex:=x+krok*cos(kierunek);           { Współrzędne końca pnia. }
  nowey:=y+krok*sin(kierunek);
  line(round(x),round(y),round(nowex),round(nowey));   { Pień. }
  drzewo(nowex, nowey, kierunek-pkmp, dalej);     { Drzewo w lewo. }
  drzewo(nowex, nowey, kierunek+pkmp, dalej)      { Drzewo w prawo. }
end;

begin
  crdmd:=Detect;
  initgraph(crdmd, crdmd, '');               { Ustaw tryb graficzny. }
{        Wywołanie drzewa...          }
  drzewo(GetMaxX div 2, GetMaxY, -pi/2, n_pokol);
  repeat until keypressed;
  closegraph
end.
```



sowania drzewa (albo jakikolwiek inny algorytm rekurencyjny). Po kilku minutach wszystko stanie się jasne.

Rekurencja, jako sposób rozwiązywania różnych zadań, jest bardzo często stosowanym w informatyce i matematyce narzędziem. Często używa się jej w różnych definicjach. Znaną Wam zapewne silnię (n! — czytaj n silnia, jest to iloczyn wszystkich liczb naturalnych mniejszych równych n i większych od zera, 0!=1 z definicji) można zdefiniować w sposób rekurencyjny:

$n!=n*(n-1)!,\ 0!=1$

Żeby obliczyć korzystając z tej definicji ile to jest 3!, musimy wykonać następujący ciąg operacji:

$3!=3*2!=3*(2*1!)=3*(2*(1*0!))=3*2*{*1}*1=6$

Podobnie jest z ciągiem Fibonacciego, którego każdy element jest sumą dwóch poprzednich:

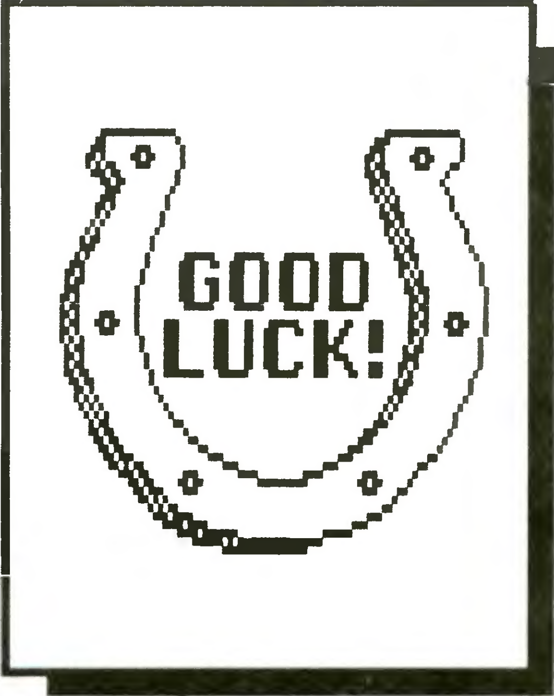
$N_i=N_{i-1}+N_{i-2},\ N_1=N_2=1$

i z wieloma innymi ciągami. Indukcja matematyczna (czyli pewien sposób przeprowadzania dowodów, znany licealistom) też jest metodą rekurencyjną. Rekurencji używa się do obliczania wartości wyrażeń algebraicznych występujących w programach — każdy składnik wyrażenia, o ile nie jest zmienną lub stałą, jest wyrażeniem, którego wartość musi zostać policzona — czyli w definicji wyrażenia występuje samo wyrażenie, co jest podstawową cechą rekurencji. Istnieją bardzo wydajne rekurencyjne metody sortowania. Przy pomocy rekurencji można także rozwiązywać zadania typu obejścia wszystkich pól na szachownicy przez skoczka, i wiele, wiele innych.

A wracając do naszego drzewa — nie uważacie, że jak na prawdziwe drzewo, wygląda w sposób nieco zbyt uporządkowany? W przyrodzie żadne drzewo nie jest idealnie symetryczne, zawsze ma którąś gałąź dłuższą, którąś krótszą, niektóre usychają wcześniej, inne później. Jak to zrealizować w naszym programie — pozostawiam Wam. Radzę użyć generatora liczb losowych, według wskazań którego można nieco modyfikować kąt między pędami, długość pnia, a także od czasu do czasu odrzucać któryś pęd jako uschnięty. Można też przyjąć, że długości wszystkich kolejnych pędów zależą od tego, do którego pokolenia należą. Efekty takich zabaw — niżej — widzicie. Spróbujcie sami się pobawić.

Marcin Borkowski





KLAN COMMODORE

PORADY SPOD LADY

„Niedawno otrzymałem w prezencie moduł o nazwie FINAL CARTRIDGE II. Niestety programy zapisane za pomocą opcji FREEZE nie chcą działać gdy zmienię ich nazwę z FC na inną. Czy wadę tą można jakoś usunąć?

Wiesiek z Elbląga

Nie jest to wada lecz typowy błąd w sztuce — prawdopodobnie źle zmieniasz nazwę programu. Zbiory zapisywane przez FINAL CARTRIDGE II i III wymagają aby druga ich część, poprzedzona znakiem „-" miała identyczny tytuł jak część pierwsza. Innymi słowy zmiana nazwy pierwszej części zbioru np. z „FC" na „TEST" wymusza jednocześnie zmianę tytułu drugiej części z „-FC" na „-TEST". Po tak dokonanych zmianach obu nazw program powinien działać bez problemów.

„Układam pewien program w języku maszynowym, który ma zarządzać zbiorami na dyskietce. Niestety bardzo często zdarza mi się, że stacja sygnalizuje błąd. Po odczytaniu kanału rozkazowego otrzymuję błąd o numerze 65 NO CHANNEL. (...)"

Błąd ten występuje zwykle podczas nieprawidłowego zarządzania wewnętrznymi buforami pamięci RAM stacji dysków oraz czasami przy odczycie zbiorów o bezpośrednim dostępie typu REL. Prawdopodobnie w tym wypadku określił Pan numer bufora wykorzystywanego przez Pana do komunikacji pomiędzy komputerem i stacją; proponowałbym jednak zostawić wybór do dyspozycji DOS poprzez zamianę numeru na znak "#".

„W programie edukacyjnym nad którym obecnie pracuję chciałem wykorzystać funkcje trygonometryczne TAN(X) i ATN(X). Jakie było zdumienie gdy okazało się, że wyniki podawane są w radianach a nie w stopniach. Jak najprościej można zamienić radiany na stopnie?"

Tak, rzeczywiście funkcje trygonometryczne są obliczane w radianach a nie w stopniach co wiąże się z systemem używanym w kraju producenta. Przeliczanie radianów na stopnie jest proste i wymaga zastosowania następującej formy obu wspominanych funkcji:

*PRINT TAN(K*PI/180): REM WYNIK JEST W STOPNIACH*

*PRINT ATN(K)*180/PI: REM WYNIK JEST W STOPNIACH*

gdzie zmienna K oznacza kąt w stopniach a zmienna PI to standardowa wartość stałej PI (3.1415926).

„Podczas wydruku ekranu graficznego stwierdziłem, że zaprojektowane przeze mnie sprite'y nie zostały przeniesione na papier. Do wydruku stosowałem program podany w BAJTKU (HARDCOPY 128) [...]"

Sprite'y nie stanowią integralnej części ekranu graficznego; dane dotyczące ich kształtów są zapisane poza obszarem pamięci w którym działa wspominany program. Dlatego też nie można ich zwykle przenieść na papier za pomocą programów tego typu.

„Czy istnieje jakiekolwiek zabezpieczenie przed źle włożonym modułem do portu EXPANSION w C-64! Mój syn omalże nie zniszczył nowego komputera wsuwając kupiony na giełdzie moduł odwrotnie [...]"

Owszem, istnieje i większość firm doskonale o tym wie, w przeciwieństwie do kilku polskich rzemieślników parających się wytwarzaniem plastikowych oprawek do modułów. Takim zabezpieczeniem jest znacznie grubsza górna krawędź obudowy modułu, co uniemożliwia jego odwrotne przyłączenie do komputera. Jak z tego widać nawet głupia obudowa powinna być przemyślana (a przynajmniej wiernie skopiowana)...

LISTA UŻYTKÓW

TYP PROGRAMU	COMMODORE 128	OSPRZET	COMMODORE 64	OSPRZET
Kalkulacyjny	SWIFTCALC 128	D	MULTIPLAN 64	D
Baza danych	dBASE II (CP/M)	D	MANAGER 64	D
Edytor tekstu	FONTMASTER 128	DP	FONTMASTER II	DP
Grafika	GRAPHIC 80	DJ	GEOPAINT	DJ(A)
Grafika 3D	CAD 3D	DJ	GIGA CAD PLUS	D
Muzyka	MUSIC MAKER 128	D	ROCK MONITOR V5.0	D
Asembler	LADS	D	CBM MACROASSEMBLER	D
System	GEOS 128	DJ(A)	GEOS V1.3	DJ(A)
Rozszerz. BASIC	METABASIC 128	D	SIMON'S BASIC	D/T
Kompilator	BASIC 128	D	BASIC 64	D
Narz. dyskowy	DOS SHELL	D	FILE COPY *	D
Kopiujacy	FAST HACK'EM V4.1B	D	FAST HACK'EM V4.1B	D
Matematyka	-		MATHEMAT	D
TURBO (tasma)	-		TURBO ROM	T
TURBO (dysk)	-		SUPERDOS V2.0	D
Modul	-		Action Replay Mk.5	-
Pakiet	GEOS	DJ(A)	GEOS	DJ(A)

* Z ZESTAWU "SUPER KIT"

SAMOPROGRAMOWANIE

mit czy rzeczywistość?

Samoprogramowanie się komputera nie jest zjawiskiem ani nowym, ani nierealnym — jest to po prostu jedna z technik programowania. Technika ta pozwala na oszczędne gospodarowanie pamięcią oraz na ukrywanie ważniejszych fragmentów programu, co z kolei stanowi jeden z elementów jego zabezpieczania lub nawet tworzenia wirusów.

Cały materiał dotyczący z konieczności podstawowych zagadnień programowania podzieliłem na trzy części, w których kolejno przedstawię dynamiczną zmianę programu napisanego w BASIC, samomodyfikowanie się programu w języku maszynowym oraz technikę znaną posiadaczom Commodore jako „dynamiczna klawiatura".

Każdy program, czy to w BASIC czy w języku wewnętrznym zapisywany jest w pamięci RAM komputera w pewnych obszarach pamięci zależnych od poszczególnych modeli. Dla interpretera BASIC w C-64 będzie to obszar od adresu 2049 do 40959, w C-16 od 4097 do 16384, w C-128 od 7169 do 65535.

Informacje dotyczące programu zapisywane są w kolejnych komórkach pamięci. Zapis ten ma zwykle następującą postać:

dwa bajty na tzw. link,
dwa bajty na numer linii programu,
n bajtów na treść linii programu,
1 bajt oznaczający zakończenie linii (zawsze 0).

Link oraz numer linii programu zapisywane są zawsze w postaci dwubajtowej, gdzie pierwszy z nich jest młodszym bajtem natomiast drugi — starszym. Zasada obliczania konkretnego adresu wskazywanego przez dwa pierwsze bajty programu lub numeru linii jest prosta: starszy bajt * 256 + młodszy bajt.

Link informuje interpreter BASIC, gdzie w pamięci rozpoczyna się następna linia tzn. następny link do następnej linii programu. Jeżeli jednak takiej linii już w programie nie ma, to w miejscu, gdzie powinien znajdować się następny link wpisane są dwa zera oznaczające zawsze koniec programu. Taki zapis przedstawiono poniżej.

2049	13	NASTEPNA LINIA OD ADRESU 8*256+13=2061
2050	8	STARSZY BAJT ADRESU NASTEPNEJ LINII
2051	10	MLODSZY BAJT NUMERU PIERWSZEJ LINII
2052	0	STARSZY BAJT NUMERU LINII (256*0+10=10)
2053	65	A
2054	178	= (DEKLARACJA ZMIENNEJ)
2055	54	6
2056	52	4
2057	55	7
2058	51	3
2059	56	8
2060	0	KONIEC LINII O NUMERZE 10
2061	21	NASTEPNA LINIA OD ADRESU 8*256+21=2069
2062	8	STARSZY BAJT ADRESU NASTEPNEJ LINII
2063	20	NUMER DRUGIEJ LINII (MLODSZY BAJT)
2064	0	NUMER DRUGIEJ LINII (STARSZY BAJT)
2065	153	TOKEN INSTRUKCJI PRINT
2066	32	SPACJA
2067	65	ZNAK ZMIENNEJ A
2068	0	KONIEC LINII O NUMERZE 20
2069	27	NASTEPNA LINIA OD ADRESU 8*256+27=2075
2070	8	STARSZY BAJT ADRESU NASTEPNEJ LINII
2071	30	NUMER TRZECIEJ LINII - MLODSZY BAJT
2072	0	STARSZY BAJT NUMERU TRZECIEJ LINII
2073	144	TOKEN INSTRUKCJI STOP
2074	0	KONIEC LINII O NUMERZE 30
2075	42	NASTEPNA LINIA OD ADRESU 8*256+42=2090
2076	8	STARSZY BAJT ADRESU NASTEPNEJ LINII
2077	40	NUMER LINII - MLODSZY BAJT
2078	0	STARSZY BAJT NUMERU LINII (0*256+40=40)
2079	151	TOKEN INSTRUKCJI POKE
2080	32	SPACJA
2081	50	2
2082	48	0
2083	54	6
2084	55	5
2085	44	,
2086	49	1
2087	53	5
2088	56	8
2089	0	KONIEC LINII O NUMERZE 40
2090	0	WRAZ Z KOMORKA 2091 OZNACZA KONIEC PROGRAMU
2091	0	

Najbardziej istotną sprawą (z puntku widzenia samoprogramowania) jest zapis poszczególnych instrukcji, funkcji i rozkazów BASIC w pamięci. Gdyby interpreter zapisywał litera po literze treść danego polecenia, to dla użytkownika i zmiennych pozostawałoby zwykle kilka bajtów pamięci; na szczęście wymyślono lepszy sposób.

Rozwiązanie jest bardzo proste — każde polecenie BASIC-a jest zapisywane w pamięci jako jeden (w C-128 niekiedy dwa) bajt, jako jedna niepowtarzalna liczba. Specjalna procedura systemu operacyjnego komputera zamienia podczas wykonywania programu tę cyfrową reprezentację danej instrukcji na samą instrukcję i następnie ją wykonuje. Dzięki temu możemy upakować w pamięci znacznie więcej rozkazów czy instrukcji w postaci jednego bajtu co oczywiście ma wpływ na wykorzystanie RAM.

Cyfrowa reprezentacja rozkazu nazywa się w języku angielskim TOKEN. Ponieważ projektanci interpretera BASIC-a dali nam do użytku instrukcję POKE, możemy zatem zmieniać bezpośrednio w pamięci dowolne token poszczególnych rozkazów i instrukcji. Przykład takiego programu przedstawiono poniżej.

```
10 A=64738
20 PRINT A
30 STOP
40 POKE 2065, 158
```

Wpisz ten program w formie przedstawionej powyżej (wraz z odstępami). Uruchom go — na ekranie zobaczysz liczbę 64738 oraz komunikat BREAK IN 30. Wpisz CONT i wciśnij RETURN. Teraz wyświetl program na ekranie za pomocą LIST.

UWAGA. Posiadacze Commodore 16/116/PLUS/4 powinni wpisać odpowiednio w linii 10 A=32793, natomiast właściciele C-128 — A=65341. Z kolei zamiast POKE 2065, 158 należy wpisać POKE 4113, 158 (C-16/116/PLUS/4) lub POKE 7185, 158 (C-128/128D). Ponadto wartości podane w wydruku poniżej mogą się różnić od siebie w pewnych komórkach (np. link).

Na ekranie zobaczymy:

```
10 A=64738
20 SYS A
30 STOP
40 POKE 2065, 158
```

Jak widać nasz program zmienił w „niepojęty sposób" instrukcję PRINT A w linii 20 na SYS A! Gdybyśmy teraz ten program uruchomili, ekran zwęzi się na chwilę i... program zniknie z pamięci — SYS 64738 oznacza wyzerowanie komputera (SYS 32793 lub 65341 to odpowiedniki dla C-16 i C-128). Wytłumaczenie tego faktu znajdziesz w tabeli — token instrukcji PRINT (liczba 153) w komórce 2065 został zastąpiony liczbą 158 czyli instrukcją SYS. Operacja ta jest realizowana w linii 40.

Jako ciekawe doświadczenie możesz wykonać (po powtórnym wpisaniu programu w takiej samej postaci) w trybie ekranowym:

```
POKE 2073, 143
```

Operacja ta zamieni z kolei instrukcję STOP na REM — zgodnie z tabelą 1.

W ten sposób możesz modyfikować dowolny program napisany w BASIC. Oczywiście instrukcje modyfikujące powinny być jak najlepiej ukryte jeżeli zależy nam na zabezpieczeniu naszego programu; radziłbym zatem stosować jak najwięcej zmiennych, „niewinnych" PRINT czy „ślepych"

INSTRUKCJA	C-64	C-128	C-16/+4
ABS	182	182	182
AND	175	175	175
APPEND	-	254 014	-
ASC	198	198	.98
ATN	193	193	193
AUTO	-	220	220
BACKUP	-	246	246
BANK	-	254 002	-
BEGIN	-	254 024	-
BEND	-	254 025	-
BLOAD	-	254 017	-
BOOT	-	254 027	-
BOX	-	225	225
BSAVE	-	254 016	-
BUMP	-	206 003	-
CATALOG	-	254 012	-
CHAR	-	224	224
CHR$	199	199	199
CIRCLE	-	226	226
CLOSE	160	160	160
CLR	156	156	156
CMD	157	157	157
COLLECT	-	243	243
COLLISION	-	254 023	-
COLOR	-	231	231
CONCAT	-	254 019	-
CONT	154	154	154
COPY	-	244	244
COS	190	190	190
DATA	131	131	131
DCLEAR	-	254 021	-
DCLOSE	-	254 015	-
DEC	-	209	209
DEF	150	150	150
DELETE	-	247	247
DIM	134	134	134
DIRECTORY	-	238	238
DLOAD	-	240	240
DO	-	235	235
DOPEN	-	254 013	-
DRAW	-	229	229
(DS)	-	+	+
(DS$)	-	+	+
DSAVE	-	239	239
DVERIFY	-	254 020	-
(EL)	-	+	+
ELSE	-	213	213
END	128	128	128
ENVELOPE	-	254 010	-
(ER)	-	+	+
ERR$	-	211	211
EXIT	-	237	237
EXP	189	189	189
FAST	-	254 037	-
FETCH	-	254 033	-
FILTER	-	254 003	-
FN	165	165	165
FOR	129	129	129
FRE	184	184	184
GET	161	161	161
(GET#)	+	+	+
(GETKEY)	-	+	+
GO	-	203	203
GOSUB	141	141	141
GOTO	137	137	137
GRAPHIC	-	222	222
GSHAPE	-	227	227
HEADER	-	241	241
HELP	-	234	234
HEX$	-	210	210
IF	139	139	139
INPUT	133	133	133
INPUT#	134	132	132
INSTR	-	212	212
INT	181	181	181
JOY	-	207	207
KEY	-	249	249
LEFT$	200	200	200
LEN	195	195	195
LET	136	136	136
LIST	155	155	155
LOAD	147	147	147
LOCATE	-	230	230
LOG	188	188	188
LOOP	-	236	236
MID$	202	202	202
MONITOR	-	250	250
MOVSPR	-	254 006	-
NEW	162	162	162
NEXT	130	130	130
NOT	168	168	168
OFF	-	254 036	-
ON	145	145	145
OPEN	159	159	159
OR	176	176	176
PAINT	-	223	223
PEEK	194	194	194
PEN	-	206 004	-
PLAY	-	254 004	-
POINTER	-	206 010	-
POKE	151	151	151
POS	185	185	185
POT	-	206 002	-
PRINT	153	153	153
PRINT#	152	152	152
PUDEF	-	221	221
QUIT	-	254 030	-
RCLR	-	205	205
RDOT	-	208	208
READ	135	135	135
RECORD	-	254 018	-
REM	143	143	143

INSTRUKCJA	C-64	C-128	C-16/+4
RENAME	-	245	245
RENUMBER	-	248	248
RESTORE	140	140	140
RESUME	-	214	214
RETURN	142	142	142
RGR	-	204	204
RIGHT$	201	201	201
RLUM	-	-	206
RND	187	187	187
RREG	-	254 009	-
RSPCOLOR	-	206 007	-
RSPPOS	-	206 005	-
RSPRITE	-	206 006	-
RUN	138	138	138
RWINDOW	-	206 009	-
SAVE	148	148	148
SCALE	-	233	233
SCNCLR	-	232	232
SCRATCH	-	242	242
SGN	180	180	180
SIN	191	191	191
SLEEP	-	254 011	-
SLOW	-	254 038	-
SOUND	-	218	218
SPC(	166	166	166
SPRCOLOR	-	254 008	-
SPRDEF	-	254 029	-
SPRITE	-	254 007	-
SPRSAV	-	254 022	-
SQR	186	186	186
SSHAPE	-	228	228
(ST)	+	+	+
STASH	-	254 031	-
STEP	169	169	169
STOP	144	144	144
STR$	196	196	196
SYS	158	158	158
SWAP	-	254 035	-
TAB(	163	163	163
TAN	192	192	192
TEMPO	-	254 005	-
THEN	167	167	167
(TI)	+	+	+
(TI$)	+	+	+
TO	164	164	164
TRAP	-	215	215
TROFF	-	217	217
TRON	-	216	216
UNTIL	-	252	252
USING	-	251	251
USR	183	183	183
VAL	197	197	197
VERIFY	149	149	149
VOL	-	219	219
WAIT	146	146	146
WHILE	-	253	253
WIDTH	-	254 028	-
WINDOW	-	254 026	-
XOR	-	206 008	-
+	170	170	170
-	171	171	171
/	173	173	173
*	172	172	172
↑	174	174	174
>	177	177	177
<	179	179	179
=	178	178	178

GOTO — zależy to już od inwencji użytkownika i twórcy programu.

Za pomocą takich POKE czy to z programu czy w trybie ekranowym możesz także zmieniać link powodując w ten sposób omijanie całych linii programu. To samo dotyczy numerów linii — wykonaj np. POKE 2050, 255 (C-64), czy POKE 4098, 255 lub POKE 7180, 255 (odpowiednio C-16 i C-128).

Pierwszy wydruk przedstawia nasz program przykładowy w formie zapisu w poszczególnych komórek pamięci. Tabela 1 natomiast ma dwie funkcje — pierwsza to informacja dotycząca token poszczególnych instrukcji i rozkazów BASIC dla 6 modeli komputerów, druga natomiast umożliwia użytkownikom porównanie trzech podstawowych wersji BASIC komputerów firmy Commodore V2.0 w C-64, V3.5 w C-16/ /116 i PLUS/4 oraz V7.0 w C-128. Instrukcje i zmienne systemowe ujęte w nawiasach (bez podanych token) podałem w tym drugim celu.

(cdn.)

Klaudiusz Dybowski

PROGRAMOWAĆ MOŻE KAŻDY

JĘZYK „C"

dla najmłodszych czyli siódmy wieczór z czarnoksiężnikiem

Dziś czarnoksiężnik czuł się nieco gorzej. Poprzedniej nocy musiał nałowić żabi skrzek i przeziębił się.

— Gdybym tak miał pomocników, którzy mogliby część pracy wykonać za mnie — rozmyślał. Wykonywałbym tylko główną część pracy. Czynności podrzędne wykonywaliby, przywoływani w odpowiednich momentach, pomocnicy. Uniknę pomyłki, jeśli będę ich przywoływał nie imieniem lecz nazwą czynności wykonywanej przez nich.

Podczas poprzednich wieczorów, gdy sam wykonywałem cały program, musiałem w nim deklarować używane przeze mnie zmienne. Pełniły one rolę narzędzi, niezbędnych do wykonania danego zadania. Analogicznie, każdy przywołany pomocnik — podprogram musi także mieć zbiór swoich narzędzi — zmiennych. Deklaracje tych zmiennych trzeba umieścić zaraz po nawiasie klamrowym rozpoczynającym jego treść. Mówimy, że są to zmienne lokalne i może z nich korzystać tylko podprogram, w którym są one zadeklarowane.

Poza tym, może się okazać, że warto pewne narzędzia — zmienne udostępnić wszystkim: majstrowi i pomocnikom. Takie zmienne nazywamy globalnymi. Ich deklaracje należy umieścić na samym początku programu głównego, przed słowem kluczowym „majster".

Każdy pomocniczy podprogram zakończy swoje działanie i odda inicjatywę programowi wywołującemu, wykonując instrukcję „oddaj". Do zbioru predef.h trzeba dopisać kolejne zaklęcie:

#define oddaj return

Instrukcja ta jest niezbędna, gdy podprogram chce razem ze sterowaniem, przekazać programowi wywołującemu wartość, będącą wynikiem jego działania. Będzie o tym jeszcze mowa.

Także wtedy, gdy podprogram chce zakończyć swoje działanie w kilku miejscach, w każdym z tych miejsc musi wykonać instrukcję „oddaj".

W przypadku, gdy nie jest przekazywana żadna wartość, użycie tej instrukcji nie jest konieczne. Podprogram odda inicjatywę także bez tej instrukcji, po napotkaniu ostatniego, zamykającego nawiasu klamrowego.

Z chwilą powrotu do programu wywołującego, giną zadeklarowane w podprogramie zmienne lokalne. W konsekwencji, są już niedostępne dla programu, który przejął dalszą inicjatywę. Różne podprogramy mogą więc używać zmiennych lokalnych o identycznych nazwach, bez obawy, że spowoduje to kolizję.

Gdybym, dla przykładu, w programie nr 3 z poprzedniego, szóstego, spotkania chciał zliczanie znaków wykonać przy pomocy podprogramu — pomocnika, to program ten mógłby wyglądać tak:

```
Program 1.

#include <stdio.h>
#include <predef.h>
 tekst t[80];
majster
{robgdy (pisz("\n Jaki tekst?"),czytajwiersz(t)>0)
        analizuj;
}
analizuj
{tekst c;
 calkowite cyfry,biale,i;
 calkowite litery,inne;
 i=litery=cyfry=biale=inne=0;
 robgdy (c=t[i++])
    gdy (c>='0' && c<='9') cyfry++;
    przeciwnie
       gdy ((c>='A' && c<='Z') ||
            (c>='a' && c<='z') litery++;
       przeciwnie
          gdy (c=='\t' || c=' ') biale++;
          przeciwnie inne++;

pisz ("\n liter jest %d",litery);
pisz ("\n cyfr jest  %d",cyfry);
pisz ("\n bialych jest %d",biale);
pisz ("\n innych jest  %d",inne);
}
```

W programie tym tablica, która przechowuje analizowany tekst, jest zmienną globalną. Ciekawostką tego programu jest pętla

robgdy [pisz ("nJaki tekst?"), czytajwiersz(t)]
analizuj;

Program będzie działał, jeśli do zbioru zaklęć ‹predef.h› dopiszemy zaklęcie:

#define czytajwiersz gets

Funkcja ta wczytuje linię tekstu do tablicy t. Jako wynik, podaje ilość wczytanych znaków. Cała pętla jest faktycznie powtarzaniem wydruku tekstu zachęty „Jaki tekst?", wczytanie tekstu do analizy i sprawdzenie czy wczytał się niepusty tekst, wreszcie ewentualna analiza wczytanego tekstu. Wydruk zachęty musi nastąpić przed wczytaniem tekstu do analizy, jednocześnie sprawdzenie warunku, czy wczytał się tekst niepusty, decyduje o wykonaniu treści pętli. Ten stan rzeczy spowodował, że w nawiasie, oprócz warunku, znalazła się instrukcja „pisz...". Obie instrukcje oddzielone są przecinkiem. Wartością wyrażenia, które składa się z kilku instrukcji oddzielonych przecinkami, jest wartość ostatniej instrukcji. W naszym przypadku jest to warunek

czytajwiersz(t)>0;

i on decyduje o kontynuowaniu programu. Podprogram „analizuj" ma tylko jed- ▶

no wyjście, zlokalizowane na jego końcu, zrezygnowano więc z instrukcji „oddaj".

Może zaistnieć sytuacja, w której wykonanie podprogramu wymaga podania pewnych parametrów. Dla przykładu — gdyby pomocnik wiercił otwory w desce, to majster musiałby mu zaznaczyć, gdzie one mają powstać oraz jaką mają mieć średnicę. Można to zrobić tak, jak w programie 1. Wartości parametrów potraktować jako zmienne globalne (ogólnodostępne). W programie tym część główna (majster) i podprogram (analizuj) korzystają z globalnej, tekstowej zmiennej t.

Taki sposób „czarowania" komputera jest bardzo nieestetyczny, może bowiem prowadzić do zbyt dużego zapełnienia pamięci i do utraty kontroli nad przepływem informacji pomiędzy poszczególnymi segmentami całego programu. Ponadto, z tak napisanych podprogramów trudniej jest utworzyć bibliotekę. Zmienne globalne trzeba deklarować jako tzw. EXTERNAL, co komplikuje obsługę takiego programu.

Wartości parametrów można przekazywać do podprogramów także w inny sposób. Wystarczy je wpisać w nawiasy okrągłe stojące przy nazwie wywoływanego podprogramu. Musi on z nich umiejętnie skorzystać. Każdy podprogram ma swoją nazwę. W jego nagłówku (nazwie) między nawiasami okrągłymi umieszczamy nazwy zmiennych będących jego parametrami. Między nazwą podprogramu a nawiasem klamrowym {, który rozpoczyna treść podprogramu, umieszczamy deklaracje tych parametrów. Za wspomnianym nawiasem klamrowym piszemy deklaracje zmiennych lokalnych i dalej treść, w której używamy zmiennych i parametrów.

W chwili wywołania podprogramu, wpisujemy w nawiasy okrągłe wyrażenia, będące aktualnymi wartościami parametrów. Oczywiście, kolejność parametrów jest istotna i musi być zgodna z deklaracją w nagłówku procedury. Dla przykładu, korzystamy z funkcji „pisz", „czytaj", „czytajwiersz" wpisując potrzebne parametry w nawiasach przy wywołaniu funkcji i więcej nic nas nie interesuje.

Spróbujmy napisać podprogram „analizuj" bez użycia zmienych globalnych, gdzie informacja będzie przekazywana jako parametry.

```
Program 2.

#include <stdio.h>
#include <predef.h>
majster
{tekst t[80];
 robgdy (pisz("\n Jaki tekst?"),czytajwiersz(t)>0)
        analizuj(t);
}
analizuj(t)
 tekst *t;
{tekst c;
 calkowite cyfry,biale,i;
 calkowite litery,inne;
 i=litery=cyfry=biale=inne=0;
 robgdy (c=t[i++])
    gdy (c>='0' && c<='9') cyfry++;
    przeciwnie
       gdy ((c>='A' && c<='Z') ||
            (c>='a' && c<='z') litery++;
       przeciwnie
          gdy (c=='\t' || c=' ') biale++;
          przeciwnie inne++;

pisz ("\n liter jest %d",litery);
pisz ("\n cyfr jest  %d",cyfry);
pisz ("\n bialych jest %d",biale);
pisz ("\n innych jest  %d",inne);
}
```

Znaczenie deklaracji:

tekst ∗t

wyjaśni się za chwilę.

Niestety, słabą stroną języka „C" jest to, że instrukcją „oddaj" możemy oddać inicjatywę programowi wywołującemu podprogram przekazując mu tylko jedną wartość. Jest to wartość wyrażenia, które umieszczamy w nawiasie przy instrukcji „oddaj". Typ tej wartości jest zazwyczaj całkowity. Jeśli jest inny, musimy to zadeklarować w nagłówku podprogramu pisząc żądany typ wyniku przed nazwą podprogramu.

Co jednak zrobić, gdy podprogram ma podać jako wynik swojej pracy, kilka wartości. Można je przekazać jako zmienne globalne, jednak wszystkie wcześniejsze uwagi pozostają w mocy. Trudność polega na tym, że zmienne lokalne przestają istnieć, gdy podprogram kończy swoją pracę. Giną więc także ich wartości.

Jest na to rada. Jako wartość parametru można podać adres zmiennej, która ma przechować wyniki. Z końcem pracy podprogram unicestwiając swoje zmienne, zniszczy adres zmiennej, przechowującej wynik a nie jej wartość.

Zmienną, która przechowuje adres innej zmiennej, nazywamy wskaźnikiem i zapisujemy ∗adres. Operator **&wart** dostarcza adres zmiennej wart. Gdy chcemy, żeby zmienna „adres" zawierała adres zmiennej „wart", wtedy musimy wykonać instrukcję

nazwa = &wart;

Po tym instrukcje: **∗nazwa = 70;** oraz **wart = 70;** wywołują ten sam skutek.

Trzeba zwrócić uwagę na wielorakie znaczenie symboli ∗ i **&**. Nie należy wpadać w panikę. Z treści programu zawsze jasno wynika, czy są to operatory iloczynów czy operacje na adresach.

Istotną jest jeszcze jedna sprawa. Otóż nazwa tablicy jest faktycznie jej adresem. Można odwołać się do treści tablicy przez wskaźnik. Właśnie dlatego w programie 2 w podprogramie „analizuj" jest deklaracja: **tekst ∗t;**

W nagłówku podprogramu, jako parametr, widnieje tylko nazwa tablicy. Faktycznie, parametrem jest adres tablicy tekstowej, musi on więc być umieszczony w zmiennej będącej wskaźnikiem.

Wskaźnik jest zmienną, może więc zmieniać swoją wartość. Do kolejnych elementów tablicy można mieć dostęp dzięki inkrementacji wskaźnika. Język „C" jest tak przyjaznym językiem, że krok tych modyfikacji jest równy rozmiarowi obiektów, z których zbudowana jest tablica. Każdorazowe wykonanie inkrementacji wskaźnika oznacza dostęp do następnego elementu tablicy, niezależnie czy jest to tablica tekstowa, czy składa się ona z liczb całkowitych, czy wreszcie z innych obiektów. Rezygnując z indeksów na rzecz inkrementacji wskaźnika zmniejszamy wprawdzie czytelność programu ale czynimy go bardziej zwartym i szybszym w działaniu.

Oto kolejna wersja naszego programu analizującego, tym razem w wersji wskaźnikowej.

```
Program 3.

#include <stdio.h>
#include <predef.h>
majster
{tekst t[80];
 robgdy (pisz("\n Jaki tekst?"),czytajwiersz(t)>0)
        analizuj(t);
}
analizuj(t)
 tekst *t;
{tekst c;
 calkowite cyfry,biale;
 calkowite litery,inne;
 litery=cyfry=biale=inne=0;
 robgdy (c=*t++])
    gdy (c>='0' && c<='9') cyfry++;
    przeciwnie
       gdy ((c>='A' && c<='Z') ||
            (c>='a' && c<='z') litery++;
       przeciwnie
          gdy (c=='\t' || c=' ') biale++;
          przeciwnie inne++;

pisz ("\n liter jest %d",litery);
pisz ("\n cyfr jest  %d",cyfry);
pisz ("\n bialych jest %d",biale);
pisz ("\n innych jest  %d",inne);
}
```

Na koniec zacytuję program, który będzie przestawiał dwie liczby. Operuje on na dwu liczbach i jego wynikiem są także dwie liczby. Musimy posłużyć się adresami zmiennych i wskaźnikami.

```
Program 4.

przestaw (px,py)
calkowite *px,*py;
{calkowite box;
 box=*px;
 *px=*py;
 *py=box;
 }
```

Wywołać tę funkcję można podając jako parametry, adresy przestawianych zmiennych, np.

przestaw(&x,&y);

przestawi wartości zmiennych x i y.

Należy podkreślić, że argumenty funkcji „czytaj" to adresy zmiennych, których wartości chcemy wczytać. Podczas drugiego, trzeciego i piątego wieczoru powstały programy, które korzystały z usług tej funkcji. Wszędzie tam, gdzie dokonuje ona wczytania wartości zmiennej liczbowej, zamiast nazwy tej zmiennej powinien być jej adres. Chochlik dokucza także czarnoksiężnikom.

Zaprezentowane dziś trzy wersje programu pokazują różne style programowania. Wprawny „czarodziej" używa wszystkich, zilustrowanych tam, technik. Każda z nich ma bowiem swoje wady i zalety. Wybór pozostawiam Wam — młodzi „czarodzieje".

Mieczysław Płacheta



ZAPIS I ODCZYT OBSZARU PAMIĘCI

Po zakończeniu „Biblioteki Action!" rozpoczynamy prezentację nadesłanych przez Czytelników procedur, które ją uzupełnią. Procedury te będą drukowane w takiej postaci, w jakiej zostały napisane przez autorów. Jednak dbałość o Czytelników wymaga od nas opatrzenia ich komentarzem.

Przede wszystkim należy powiedzieć, że obie pokazane procedury można zastąpić przez opisane w „Bajtku" 4/89 procedury **BGet()** i **Bput()**. Zwróćmy jednak uwagę na ich budowę. Ponieważ nie można użyć procedury w wywołaniu procedury, więc konieczne było wprowadzenie pomocniczej zmiennej **w.** Zwracałem jednak uwagę na zbędność procedur **Peek()** i **Poke()** w **Action!** Zamiast nich wystarczy odpowiednia deklaracja zmiennych. W procedurze **Save()** należy zadeklarować tablicę

BYTE ARRAY w (d1)=ad

i dzięki temu wewnątrz pętli **FOR** wystarczy instrukcja

PutD(1,w(i))

Analogicznie można postąpić w drugiej procedurze.

Procedura **Save()** zawiera ponadto błąd, który jest na pierwszy rzut oka niewidoczny, a nawet nie przeszkadza w jej działaniu. Ujawnia się on dopiero po dokonaniu pokazanej wyżej zmiany. Otóż pętla **FOR** zapisuje na taśmę obszar pamięci od adresu **ad** do adresu **ad+d1.** Czyli z komórki **ad** pobiera pierwszy bajt, z komórki **ad+1** drugi itd. Dla przykładu użytego w **TestSL()** przedostatni bajt będzie pobrany z **ad+3199,** a ostatni z **ad+3200.** Razem daje to 3201 bajtów, a więc o jeden za dużo. Po prostu pierwszy element ma numer 0! Należy o tym pamiętać i w podobnych przypadkach zmniejszać górną granicę pętli o 1.

(W.Z.)

Pierwsza procedura umożliwia zapisanie na taśmie zawartości pamięci komputera od wskazanego adresu i o zadanej długości (w bajtach). Jest to procedura **Save().**

Druga procedura nosi nazwę **Load()** i jest niejako sprzężona z pierwszą, gdyż umożliwia odczyt danych z magnetofonu i ponowne umieszczenie ich w pamięci komputera od dowolnie wskazanego miejsca.

Procedura **TestSL()** demonstruje, w jaki sposób można wykorzystać opisane wyżej procedury do zapisu na taśmie zawartości obrazu w trybie 7, a następnie do jej odczytania (z tejże samej taśmy).

Procedurę **Save()** należy wywoływać podając dwa parametry, czyli Save (pierwszakomórka, długośćbloku), gdzie „pierwszakomórka" to adres komórki pamięci, od której będziemy zapisywać, a „długośćbloku" to liczba zapisywanych bajtów. W przypadku **Load()** podajemy tylko jeden parametr, który wskazuje komórkę, od której pamięć ma być zapełniana.

Andrzej Postrzednik

```
; Zapis i odczyt RAM
; Andrzej Postrzednik
; Copyright (c) Bajtek

; zapis na tasme

PROC Save(CARD ad,dl)
 BYTE w
 CARD i

 Close(1)
 Open(1,"C:",8,128)
 FOR i=ad TO ad+dl
  DO
   w=Peek(i)
   PutD(1,w)
  OD
 Close(1)
RETURN

; odczyt z tasmy

PROC Load(CARD ad)
 BYTE w
 CARD i

 Close(1)
 Open(1,"C:",4,128)
 i=ad
 DO
  w=GetD(1)
  Poke(i,w)
  i==+1
  UNTIL Peek(63)<>0
 OD
 Close(1)
RETURN

; zapis i odczyt
; obrazu w trybie 7

PROC TestSL()
 CARD pk=88

 Graphics(7) color=1
 Plot(0,0)
 Drawto(150,70)
 Drawto(70,5)
 PrintE("ZAPIS")
 Save(pk,40*80)
; 40*80 = rozmiar pamieci
; obrazu w trybie 7
 Graphics(7)
 PrintE("ODCZYT")
 Load(pk)
RETURN
```

OBROŃCA ZIEMI

Już od dwóch lat trwa zacięta wojna pomiędzy mieszkańcami Neptuna i Ziemi — Twojej ojczyzny.

Jesteś dowódcą patrolowego kosmolotu, który tkwi w przestrzeni, międzyplanetarnej uszkodzony w wyniku drobnej potyczki. Cała ziemska flota znajduje się po drugiej stronie Układu Słonecznego, a na Twoim radarze ukazał się statek przeciwnika transportujący na jedną z planetoid ładunek wybuchowy o ogromnej sile niszczącej.

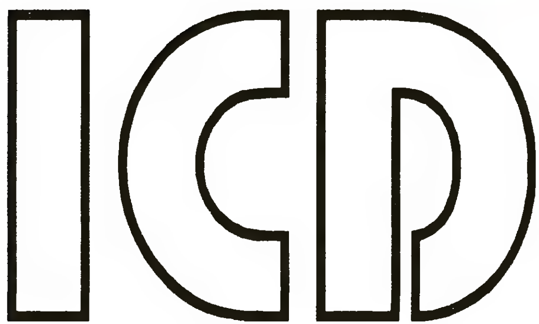
Twoim zadaniem jest zniszczenie tego statku, co wymaga jego czterokrotnego trafienia. Możesz liczyć tylko na własne siły, gdyż radiostacja nie działa i nie można wezwać innych kosmolotów na pomoc. Niestety w wyniku uszkodzeń Twój okręt nie może się także poruszać, a na domiar złego w magazynie amunicyjnym pozostało już tylko 10 pocisków.

Na Twoich barkach spoczywa teraz los Ziemi. Jeżeli dopuścisz do zgromadzenia na planetoidzie materiału wybuchowego, to przeciwnik zniszczy Twoją planetę. Jeżeli zabraknie Ci amunicji, stracisz swój okręt. Pamiętaj! Od tej walki zależy los całej Ziemi.

Gdy przegrasz, wystarczy pociągnąć joystick do siebie, aby gra rozpoczęła się od nowa.

Piotr Karpiuk

```
NY   10 REM OBRONCA ZIEMI
JE   20 REM Piotr Karpiuk
MY   30 REM (c) 1990, Bajtek
BB   40 REM
NQ   70 GRAPHICS 15:COLOR 2:SETCOLOR 2,0,0
EP   72 DE=INT(158*RND(0))
AZ   74 FG=INT(140*RND(0))
CR   76 PLOT DE,FG:DRAWTO DE,FG+1
YI   80 Z=25:P=10:M=0:F=0:AB=3:GH=25:BD=0
LJ   81 REM *** RYSOWANIE GWIAZD ***
HL   82 J=INT(156*RND(0))
KB   84 T=INT(158*RND(0))
NM   86 PLOT J,T:DRAWTO J,T+1
AE   88 M=M+1
SM   90 IF M=30 THEN 110
UZ   92 GOTO 82
GG   110 REM *** RYSOWANIE POJAZDU ***
VO   120 READ A,B,C
FN   130 PLOT A,B:DRAWTO C,B
AE   140 IF A=0 THEN 1000
AF   150 DATA 7,50,13,13,51,14,14,52,15,15,
     53,16,6,54,18,6,55,19,6,56,19
ZF   160 DATA 6,57,18,7,51,7,7,52,7,7,53,7,
     6,54,6,6,56,6,6,55,6,0,0,0
MU   162 GOTO 120
BM   1000 REM *** RUCH WROGA ***
SF   1005 SOUND 0,255,8,6
QD   1010 COLOR 0:PLOT 0,0:DRAWTO 0,0
EE   1020 X=158:Q=30
SJ   1025 COLOR 2:PLOT X,28:DRAWTO X,29:PLO
     T X,99:DRAWTO X,100
DY   1030 COLOR 2:PLOT X,Q:DRAWTO X,Q
AN   1035 IF STRIG(0)=0 THEN 2000
OX   1040 Q=Q+1
QY   1046 GOTO 1060
XI   1050 ? CHR$(125):? "Na planetoidzie je
     st zgromadzone ";P;"%  materialu wybuc
     howego."
IY   1051 P=P+10
MS   1052 IF P=110 THEN 5000
PN   1059 GOTO 1020
MH   1060 COLOR 0:PLOT X,Q-1:DRAWTO X,Q-1
NR   1070 IF Q=100 THEN 1050
PC   1090 GOTO 1030
FK   2000 REM *** STRZELANIE ***
WP   2010 FOR I=0 TO 20:SOUND 1,I,12,15:NEX
     T I:SOUND 1,0,0,0
UT   2015 H=21:I=55:J=159
LS   2020 COLOR 2:PLOT H,I:DRAWTO J,I
LC   2022 COLOR 0:PLOT H,I:DRAWTO J,I
BG   2025 IF J=159 AND Q=55 THEN 3000
UA   2026 REM *** LICZENIE STRZALOW ***
JW   2027 F=F+1
RX   2028 IF F=10 THEN 8000
QD   2030 ? CHR$(125):? "liczba pociskow: "
     ;10-F
OR   2050 GOTO 1030
ZX   3000 FOR L=15 TO 0 STEP -1:SOUND 0,240
     ,0,L:SOUND 1,245,0,L:SOUND 2,250,2,L:S
     OUND 3,255.4.L:FOR T=0 TO 30
UQ   3010 NEXT T:NEXT L:? CHR$(125):? "uszk
     odzenia wroga: ";Z;"%"
RB   3015 Z=Z+25
RU   3020 IF Z=125 THEN 4000
FJ   3025 Q=56:COLOR 0:PLOT 158,55:DRAWTO 1
     58,55
PX   3030 GOTO 2050
EA   4000 REM *** KONIEC (WYGRANY) ***
TQ   4002 GRAPHICS 1+16
FP   4005 POSITION 4,11:? #6;"wygrales!!!"
WI   4010 FOR S=0 TO 90:NEXT S:POSITION 4,1
     1:? #6;"                   ":FOR S=0 TO 90:
     NEXT S
QW   4020 GOTO 4005
EH   5000 REM *** KONIEC (PRZEGRANY) ***
KF   5001 FOR I=60 TO 30 STEP -1:SOUND 1,I,
     12,15:NEXT I:SOUND 1,0,0,0
UY   5010 COLOR 2:PLOT 158,100:DRAWTO DE,FG
UU   5015 COLOR 0:PLOT 158,100:DRAWTO DE,FG
AF   5020 FOR L=15 TO 0 STEP -1:SOUND 0,240
     ,0,L:SOUND 1,245,0,L:SOUND 2,250,2,L:S
     OUND 3,255,4,L:FOR T=0 TO 30
ND   5030 NEXT T:NEXT L:GRAPHICS 1+16
JR   5040 POSITION 3,10:? #6;"twoja planeta
     "
MB   5060 POSITION 3,12:? #6;"zniszczona!!!
     "
BE   5065 IF STICK(0)=13 THEN RUN
YX   5070 FOR S=0 TO 90:NEXT S:POSITION 3,1
     2:? #6;"                   ":FOR S=0 TO 90:
     NEXT S:GOTO 5060
JC   8000 FOR S=0 TO 300:NEXT S:? "Nie masz
      juz pociskow":GOTO 20000
HF   10000 REM *** KONIEC (PRZEGRANY) ***
SR   10001 GRAPHICS 1+16:POSITION 4,11:? #6
     ;"przegrales!!!":FOR S=0 TO 90:NEXT S
PC   10005 IF STICK(0)=13 THEN RUN
XW   10010 POSITION 4,11:? #6;"
      ":FOR S=0 TO 90:NEXT S:GOTO 10000
XC   20000 REM * KONIEC (BRAK POCISKOW) *
VH   20001 FOR K=1 TO 70:SOUND 0,K,8,15:NEX
     T K:SOUND 0,0,0,0
PN   20010 COLOR 2:PLOT 158,Q:DRAWTO 17,53
OV   20030 COLOR 0:PLOT 158,Q:DRAWTO 17,53
SE   20040 ? CHR$(125):? "Twoje uszkodzenia
     : ";GH;"%"
VM   20050 GH=GH+25
MK   20060 BD=BD+1
OE   20070 IF BD=4 THEN 20500
WZ   20080 GOTO 20000
KK   20500 GRAPHICS 17:FOR L=15 TO 0 STEP -
     1:SOUND 0,240,0,L:SOUND 1,245,0,L:SOUN
     D 2,250,2,L:SOUND 3,255,4,L
DY   20510 FOR T=0 TO 30:NEXT T:NEXT L
VA   20530 POSITION 5,10:? #6;"twoj statek"
YS   20540 POSITION 5,12:? #6;"zniszczony!"
VQ   20550 FOR S=0 TO 500:NEXT S
WU   20560 GOTO 10000
```



DLA ATARI

Użytkownikom Atari dobrze znana jest amerykańska firma OSS. Pochodzi z niej większość języków programowania oraz wiele systemów operacyjnych. Znacznie mniejszą popularnością cieszy się w Polsce firma ICD, która zajmuje się produkcją sprzętu peryferyjnego.

W styczniu 1988 roku oba te przedsiębiorstwa połączyły się i oferują swoje produkty pod wspólną marką ICD. A są to produkty nie byle jakie. Opisywałem już jeden z nich — SpartaDOS X („Bajtek" 11/89). Chciałbym więc teraz przedstawić Czytelnikom pełną ofertę firmy ICD na rok 1990 dla ośmiobitowych komputerów Atari.

P:R:Connection — interfejs dołączany do złącza szeregowego. Posiada dwa porty RS-232 i jeden port Centronics, a ponadto dodatkowe złącze szeregowe. Zasilany jest z komputera i ma wbudowaną procedurę obsługi urządzenia „R:". Dodatkowe oprogramowanie dostarczane jest na dyskietce.

Printer Connection — interfejs Centronics, zasilany z komputera i wyposażony w przewód o długości 3 m. Znacznie tańszy jest jednak MicroPrint (dostępny w Pewexie).

RAMBO XL — rozszerzenie pamięci do 256 KB dla 800XL i 1200XL w pełni zgodne ze 130XE, Basic XE i SpartaDOS. Instalacja w komputerze wymaga posiadania wkrętaka i trwa kilka minut.

Multi I/0 — wielofunkcyjny interfejs z odrębnym zasilaniem, przyłączany do szyny równoległej. Realizuje pięć niezależnych funkcji:

— RAMdysk o pojemności 256 KB lub 1 MB (zależnie od wersji), który jest niezależny od pamięci komputera, a dzięki oddzielnemu zasilaniu nie traci zawartości po wyłączeniu komputera;

— standardowy interfejs Centronics o zmiennej konfiguracji, działa jako P1: lub P2: z wyłączaną funkcją *auto line feed*;

— interfejs szeregowy RS-232 z wbudowaną procedurą obsługi urządzenia R: (oszczędza prawie 2 KB pamięci);

— bufor drukarki w pamięci RAM z możliwością zatrzymania i wznowienia druku oraz wydrukowania kopii;

— interfejs standardu SCSI/SASI umożliwiający przyłączenie ośmiu twardych dysków.

Dodatkowe oprogramowanie umożliwia ustalenie konfiguracji pracy interfejsu, odczyt wstępny (po włączeniu komputera) z RAMdysku lub twardego dysku oraz podział twardego dysku na partycje o pojemności 16 MB i instalację ośmiu RAMdysków w pamięci RAM.

Hard Drive Kit — zestaw do montażu dowolnego napędu twardego dysku 5,25'. Zawiera zasilacz 60 W, kontroler dysku, obudowę i wszystkie niezbędne przewody połączeniowe. Napęd twardego dysku należy jednak nabyć osobno, a ponadto konieczne jest posiadanie interfejsu Multi I/0.

SpartaDOS Construction Set — zestaw konstrukcyjny SpartaDOS na dwóch dyskietkach. Pozwala na stworzenie własnego systemu opartego na dowolnej wersji SpartaDOS (2.3, 3.2 lub X). W zestawie zawarte są programy umożliwiające współpracę ze wszystkimi urządzeniami dostępnymi dla Atari (m.in. dyski twarde, US Doubler, ATR8000, RAMdysk do 1 MB).

SpartaDOS X — dyskowy system operacyjny w module RAM (cartridge). Szerszy opis był zamieszczony w „Bajtku" 1/90.

SpartaDOS ToolKit — dodatkowe oprogramowanie dla SpartaDOS: program sortujący katalogi dyskietek, edytor dyskowy, programator klawiszy i wiele innych programów użytkowych.

FlashBack! — specjalny program kopiujący dla twardego dysku — pozwala na zapisanie jego zawartości na zwykłych dyskietkach. W pełni zautomatyzowany — wymaga tylko wymieniania dyskietek. Pliki są wybierane automatycznie (nigdy nie kopiuje dwa razy tego samego pliku).

US Doubler — rozszerzenie stacji dysków Atari 1050 pozwalające na uzyskanie podwójnej gęstości zapisu i transmisji z szybkością 54 000 bodów. Instalowany samodzielnie przez użytkownika.

R-Time 8 — moduł (cartridge) zegara czasu rzeczywistego z baterią zapewniającą działanie przez pięć lat. Posiada gniazdo do włożenia następnego modułu (np. SpartaDOS X, Action! itd.) i jest automatycznie rozpoznawany przez SpartaDOS.

MAC/65 — moduł ROM zawierający asembler MAC/65 oraz debugger DDT. Jest to jeden z najlepszych asemblerów dla Atari. Jego opis był opublikowany w „Bajtku" 3/89.

MAC/65 ToolKit — dyskietka z zestawem dodatkowych procedur upraszczających tworzenie programów w asemblerze MAC/65. Zawiera m.in. procedury OPEN, CLOSE, PRINT, GOSUB, IF i DO oraz liczne procedury graficzne, w tym również do obsługi grafiki graczy i pocisków oraz delikatnego przesuwu obrazu.

Action! — moduł ROM zawierający język programowania Action!. Bardzo prosty w nauce i rewelacyjnie szybki język (tylko 1,5 raza wolniejszy od Turbo Pascala na IBM). Moduł pozostawia ponad 30 KB pamięci dla użytkownika (wersja dyskowa tylko 16 KB).

Action! ToolKit — dyskietka z zestawem dodatkowych procedur upraszczających tworzenie programów w języku Action!. Zbliżone procedury były już kilkakrotnie publikowane w „Bajtku".

Action! RunTime Package — dyskietka z programem i biblioteką umożliwiającymi uruchomienie programów napisanych w Action! bez modułu.

Basic XL — moduł ROM zawierający język programowania Basic XL, znacznie rozbudowany w stosunku do Atari Basic. Krótki opis tego języka był opublikowany w „Bajtku" 10/87.

Basic XL ToolKit — dyskietka z zestawem dodatkowych procedur upraszczających tworzenie programów w języku Basic XL. Zawiera ponadto *„RunTime Package"*, który pozwala na uruchamianie programów w Basicu XL bez modułu.

Basic XE — moduł ROM zawierający język programowania Basic XE, zaprojektowany specjalnie dla komputerów 130XE i wykorzystujący całą ich pamięć (128 KB). Krótki opis tego języka był opublikowany w „Bajtku" 11/87.

Wojciech Zientara

TRÓJWYMIAROWE WYKRESY

Rysowanie trójwymiarowych wykresów funkcji dwóch zmiennych wcale nie wymaga tworzenia skomplikowanych programów. Przyjmując pewne założenia można to zrealizować nawet przy pomocy prostego programu w BASIC-u.

Prezentowany tu program jest właśnie takim prostym rozwiązaniem. Nie wprowadziłem w nim zmiennych zakresów, ani usuwania niewidocznych linii. Przyjęte zakresy wynikają z rozdzielczości obrazu (tryb 8+16) i wynoszą x (−104;106), y (−30;30) i z (−48;48). Nie stanowi to żadnej przeszkody, ponieważ we wzorze funkcji wystarczy dobrać odpowiednie współczynniki i wykres funkcji będzie wypełniał ekran. Dzięki temu, że tylko co ósmy płaski wykres jest rysowany w całości, cały program działa w dosyć sensownym czasie.

Po zmianie DEG w wierszu 50 na RAD wartości funkcji trygonometrycznych będą obliczane w radianach. Właściwą funkcję należy umieścić w wierszu 290. Oto kilka przykładowych funkcji:

Y=5E−3∗X∗Z−0.01∗Z∗Z∗SIN(4∗X)
Y=0.6∗X∗SIN(2.4∗X)∗SIN(2∗Z)
Y=20∗SIN(5∗X)∗SIN(Z)

oraz funkcje rysowane wolniej, lecz ciekawsze:

Y=−10∗SIN(2∗Z)−10∗COS(2∗X)+10∗COS(X/9∗Z)
Y=−10∗SIN(2∗X)−10∗COS(2∗Z)+10∗SIN(X/9∗Z)

Andrzej Orłowski

```
GN   10 REM WYKRES 3D
VP   20 REM Andrzej Orlowski
MY   30 REM (c) 1990, Bajtek
OS   40 GRAPHICS 24
TJ   50 POKE 710,0:COLOR 1:DEG
MF   60 FOR Z=-48 TO 48
RW   70 IF INT(Z/8)=Z/8 THEN GOSUB 200
YM   80 IF INT(Z/8)<>Z/8 THEN GOSUB 140
OU   90 NEXT Z
WH   100 POKE 764,255
YM   110 IF PEEK(764)=255 THEN 110
NL   120 LIST 290:END
VC   130 REM * PUNKTY *
PU   140 FOR X=-104 TO 106 STEP 10
UZ   150 GOSUB 280
CZ   160 PLOT X-Z+160,-Y+Z+88
LZ   170 NEXT X
ZN   180 RETURN
WR   190 REM * LINIA *
HL   200 X=-104:GOSUB 280
CQ   210 PLOT X-Z+160,-Y+Z+88
TF   220 FOR X=-103 TO 106
UW   230 GOSUB 280
YY   240 DRAWTO X-Z+160,-Y+Z+88
LW   250 NEXT X
ZK   260 RETURN
RN   270 REM * FUNKCJA *
LB   280 TRAP 300
JR   290 Y=20*SIN(X*5)*COS(Z*4)
ME   300 IF Y>30 THEN Y=30
SN   310 IF Y<-30 THEN Y=-30
ZD   320 RETURN
```

Na życzenie Czytelników również w klanie Atari będziemy publikować listę najlepszych i najpopularniejszych programów użytkowych. W przypadku komputerów Atari różnice pomiędzy poszczególnymi modelami (800XL, 800XE, 65XE i 130XE) są mało istotne, a znacznie ważniejszy jest rodzaj stosowanej pamięci masowej. Z tego względu programy zostały podzielone na dwie kategorie: przeznaczone do współpracy ze stacją dysków i z magnetofonem. Nieco inny jest układ samej tabeli, lecz może on jeszcze ulec zmianie, jeśli będą tego wymagały nadesłane przez Czytelników propozycje.

(ziew)

LISTA UŻYTKÓW DLA ATARI

TYP PROGRAMU	STACJA DYSKÓW	MAGNETOFON
Kalkulacyjny	VisiCalc	
Baza danych	SynFile+	
Edytor tekstu	First XLEnt	SpeedScript
Pakiet biurowy	Mini Office 2	
Grafika (artyst.)	RAMbrandt	Koala
Grafika (techn.)	Design Master	
Muzyka	Sound Machine	Sound Machine
Matematyka	Calculator	
System	Bibo DOS 5.4	KOS 2.1
Asembler	MAC/65	MAC/65
Monitor	BUG/65	BUG/65
Basic	Turbo-Basic	Turbo-Basic
Kompilator Basica	MMG Compiler	
Inny język	Action!	Action!
Narzędziowy	Disk Wizard 2	
Kopiujący	Track Copier	CasDup 2.0
Rozszerz. sprzętu	Happy Warp	A Super Turbo

80 ZNAKÓW W WIERSZU

Pod takim tytułem w „Bajtku" 10/88 opublikowany został program umożliwiający otrzymanie 80 znaków w każdym wierszu ekranu. Przedstawiam teraz jego znacznie ulepszoną wersję — krótszą, szybszą i wygodniejszą w użyciu.

Wydrukowany obok program zawiera tylko procedurę w języku maszynowym — bez matrycy znaków. Aby uzyskać jego poprawne działanie, należy najpierw wczytać i uruchomić pierwszą wersję programu, a dopiero potem drugą (opublikowaną w tym numerze). Można, oczywiście, połączyć oba programy dla otrzymania jednolitej wersji.

Przed wykorzystaniem procedury we własnym programie konieczne jest otwarcie urządzenia instrukcjami

GRAPHICS 8:OPEN #1,0,0,"G:"

Teraz można już umieścić na ekranie dowolną informację przy pomocy instrukcji

POSITION X,Y:? #1;A$

lub

POSITION X,Y:? #1;A

Znaczenie zmiennych użytych w tych instrukcjach jest takie samo, jak w pierwotnej wersji programu, więc użytkownicy, którzy wykorzystali ją we własnych programach, muszą tylko zmienić procedurę maszynową.

Program w aktualnej wersji posiada możliwość ustawienia marginesów: lewego w komórce 207 i prawego w komórce 208. Wartość prawego marginesu musi być przy tym zwiększona o 1. Jeżeli wyświetlany tekst wyjdzie poza prawy margines, to jego pozostała część zostanie przeniesiona do następnego wiersza. Przecinki i średniki między drukowanymi zmiennymi mają identyczne działanie, jak w trybie GRAPHICS 0. Pozostałe funkcje programu pozostały niezmienione, lecz zamiast:

POKE 1770,128

należy stosować:

POKE 1655,121

a zamiast:

POKE 1664,13

wykorzystuje się:

POKE 1657,13

Normalny tryb pracy programu przywracają instrukcje:

POKE 1655,124:POKE 1657,77

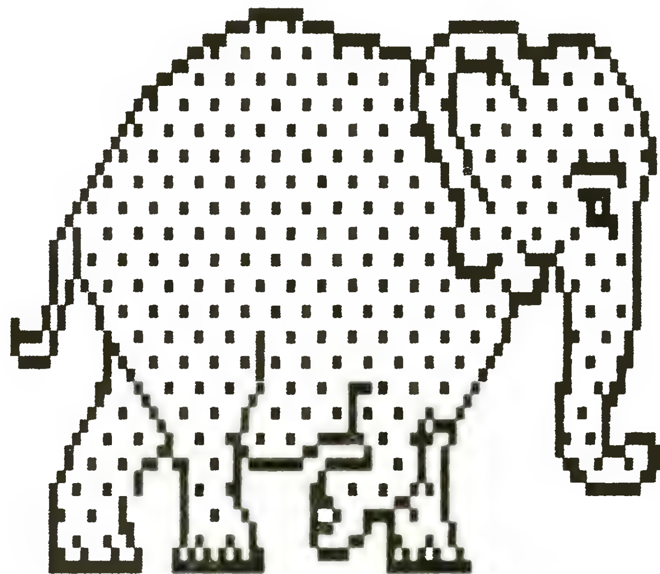
Michał Skowroński

LISTING 1

```
NZ   0 REM 80 ZNAKOW - WERSJA 2
OI   1 REM Michal Skowronski
HZ   2 REM (c) 1990, Bajtek
NI   3 REM
FJ   1000 FOR A=1536 TO 1730
WE   1010 READ X:POKE A,X
BL   1020 NEXT A
IQ   1030 POKE 812,71:POKE 813,0:POKE 814,6
BJ   1040 POKE 207,0:POKE 208,80
FC   1050 END
YM   2000 DATA 157,6,157,6,157,6,11,6,157,6
OJ   2010 DATA 157,6,72,201,155,208,7,32,18
     3,6
HO   2020 DATA 160,1,104,96,165,85,197,208,
     208,3
RW   2030 DATA 32,183,6,104,133,203,169,8,1
     33,204
NZ   2040 DATA 32,161,6,169,0,24,101,205,14
     1,110
TK   2050 DATA 6,169,124,101,206,141,111,6,
     165,84
YA   2060 DATA 133,203,169,40,133,204,32,16
     1,6,165
GA   2070 DATA 85,74,8,24,101,205,133,205,1
     69,0
QH   2080 DATA 101,206,133,206,165,205,24,1
     01,88,141
UQ   2090 DATA 122,6,141,125,6,165,206,101,
     89,141
DU   2100 DATA 123,6,141,126,6,160,0,40,8,1
     85
RW   2110 DATA 72,126,176,7,10,10,10,10,76,
     124
VG   2120 DATA 6,13,210,131,141,210,131,173
     ,122,6
DS   2130 DATA 24,105,40,141,122,6,141,125,
     6,173
TH   2140 DATA 123,6,105,0,141,123,6,141,12
     6,6
LB   2150 DATA 200,192,8,208,208,104,230,85
     ,160,1
NG   2160 DATA 96,169,0,133,205,162,8,70,20
     3,144
PI   2170 DATA 3,24,101,204,106,102,205,202
     ,208,243
VH   2180 DATA 133,206,96,165,207,133,85,24
     ,169,8
QP   2190 DATA 101,84,133,84,96
```

LISTING 2

```
0100 ; 80 ZNAKOW - WERSJA 2
0110 ; Michal Skowronski
0120 ; (c) 1990, Bajtek
0130 ;
0140 MN1  =    203
0150 MN2  =    204
0160 WMNL  =   205
0170 WMNH  =   206
0180 ;
0190      *=   1536
0200 ;
0210      .WORD KO-1,KO-1,KO-1
0220      .WORD PU-1,KO-1,KO-1
0230 ;
0240 PU   PHA
0250      CMP #155      ;RETURN ?
0260      BNE S1
0270      JSR NL
0280      LDY #1
0290      RTS           ;POWROT
0300 S1   LDA 85        ;poza
0310      CMP 208       ;marginesem ?
0320      BNE S2
0330      JSR NL
0340 S2   PLA
0350      STA MN1       ;pomnozenie
0360      LDA #8
0370      STA MN2       ;przez 8
0380      JSR PMN
0390      LDA #0
0400      CLC
0410      ADC WMNL      ;dodanie
0420      STA AP+1      ;adresu
0430      LDA #124      ;matrycy
0440      ADC WMNH      ;znakow
0450      STA AP+2
0460      LDA 84        ;pomnozenie
0470      STA MN1       ;pozycji
0480      LDA #40       ;pionowej
0490      STA MN2       ;przez 40
0500      JSR PMN
0510      LDA 85
0520      LSR A
0530      PHP           ;dodanie
0540      CLC
0550      ADC WMNL
0560      STA WMNL      ;pozycji
0570      LDA #0
0580      ADC WMNH
0590      STA WMNH      ;poziomej
0600      LDA WMNL
0610      CLC
0620      ADC 88        ;dodanie
0630      STA AZ1+1
0640      STA AZ2+1     ;adresu
0650      LDA WMNH
0660      ADC 89        ;pamieci
0670      STA AZ1+2
0680      STA AZ2+2     ;obrazu
0690      LDY #0
0700      PLP
0710      PHP
0720 AP   LDA 1000,Y    ;pobranie
0730      BCS AZ1       ;nieparzysta ?
0740      ASL A
0750      ASL A
0760      ASL A         ;przesuniecie
0770      ASL A
0780      JMP AZ2
0790 AZ1  EOR 1000
0800 AZ2  STA 1000      ;wpisanie
0810      LDA AZ1+1
0820      CLC
0830      ADC #40       ;zwiekszenie
0840      STA AZ1+1
0850      STA AZ2+1     ;o 40
0860      ADC #0
0870      STA AZ1+2
0880      STA AZ2+2
0890      INY
0900      CPY #8        ;wykonaj
0910      BNE AP        ;8 razy
0920      PLA
0930      INC 85        ;zwieksz. poz X
0940 KO   LDY #1
0950      RTS           ;POWROT
0960 PMN  LDA #0        ;podprogram
0970      STA WMNL      ;mnozenia
0980      LDX #8
0990 PE   LSR MN1
1000      BCC S3
1010      CLC
1020      ADC MN2
1030 S3   ROR A
1040      ROR WMNL
1050      DEX
1060      BNE PE
1070      STA WMNH
1080      RTS           ;powrot z PMN
1090 NL   LDA 207       ;przeniesienie
1100      STA 85        ;do nastepnej
1110      CLC           ;linii
1120      LDA #8
1130      ADC 84
1140      STA 84
1150      RTS           ;powrot z NL
1160      .END
```



KASETOWY RAMDYSK

W „IKS-ie” 1/87 został opublikowany program H. Krasuskiego, tworzący dodatkowe urządzenia zewnętrzne. Poniżej publikujemy jego niewielką modyfikację. Zmiana polega na dodaniu do oryginalnej procedury kilku liczb, dzięki czemu ramdysk może być uruchamiany z poziomu DOS-u.

Program ten dodaje do systemu Atari dwa nowe urządzenia zewnętrzne:
M: — ramdysk (zapis i odczyt),
N: — nic (tylko zapis, dane nie są nigdzie składowane).
Większość programów akceptuje tą zmianę i możemy w nich używać tych urządzeń. Jest to szczególnie przydatne dla posiadaczy magnetofonów.

Ta wersja ramdysku powstała m.in. jako odpowiedź na list Rolanda Zimka dotyczący problemu utraty programu w **Action!** po wciśnięciu ‹RESET›. Ramdysk pozwala na odzyskanie programu napisanego w **Action!**, pod warunkiem, iż nagramy go wcześniej na urządzenie M:. Trwa to najwyżej 3 sekundy, a więc nic nas nie kosztuje. Gdy nastąpi konieczność naciśnięcia ‹RESET›, możemy to zrobić bez obawy. Wystarczy potem uruchomić ramdysk (adres 1536 lub program uruchamiacz) i wczytać **Action!**, a następnie odczytać do edytora program z urządzenia M:.

W przypadku posiadania magnetofonu sytuacja jest podobna. Przed wczytaniem pliku **Action!** należy zamiast standardowego „!” wczytać np. BLC i na niego ramdysk, a dopiero potem **Action!**. Gdy użyjemy ‹RESET›, wczytujemy „Uruchamiacz” i znów **Action!**. Utracony program można odczytać z urządzenia M:.

Sebastian Siwy

```
AT   1 REM * RAMDYSK *
OK   2 REM Sebastian Siwy
DE   3 REM modyfikacja programu
RO   4 REM H. Krasuckiego, IKS 1/87
IC   5 REM (c) 1990, Bajtek
NL   6 REM
TV   10 DIM N$(20)
YR   20 RESTORE :FOR I=0 TO 180:READ A:S=S+
     A:NEXT I
ZF   30 IF S<>18742 THEN ? "BLAD !!":END
DI   40 ? "Podaj specyfikacje pliku RAMDYSK
     U":INPUT N$
QJ   50 OPEN #1,8,255,N$
QP   60 RESTORE 100:FOR I=0 TO 164:READ A:P
     UT #1,A:NEXT I:CLOSE #1
KH   70 ? "Podaj specyfikacje pliku URUCHAM
     IACZA":INPUT N$
QM   80 OPEN #1,8,255,N$
LD   90 RESTORE 200:FOR I=0 TO 15:READ A:PU
     T #1,A:NEXT I:CLOSE #1
CY   95 ? "Programy nagrano jako pliki DOS-
     u."
FQ   99 REM RAMDYSK
JT   100 DATA 255,255,0,6,235,6,160,0,185,2
     6,3,201,0,240,9,200,200,200,192,34
FS   110 DATA 208,242,56,96,162,0,189,62,6,
     153,26,3,232,200,224,6,208,244,174,0
XQ   120 DATA 160,232,138,142,0,160,77,0,16
     0,208,4,206,0,160,96,169,160,141,99,6
KF   130 DATA 141,235,6,96,104,76,0,6,78,68
     ,6,77,83,6,105,6,105,6,107,6
AT   140 DATA 105,6,105,6,107,6,76,106,6,97
     ,6,105,6,160,6,108,6,105,6,105
BV   150 DATA 6,76,106,6,169,192,133,75,169
     ,0,133,74,160,1,96,162,208,228,75,208
TG   160 DATA 4,162,216,134,75,160,0,196,75
     ,240,28,32,209,6,145,74,32,224,6,230
HS   170 DATA 74,196,74,208,2,230,75,200,16
     5,74,141,234,6,165,75,141,235,6,96,32
VO   180 DATA 98,6,160,146,76
CX   199 REM URUCHAMIACZ
WV   200 DATA 255,255,0,144,3,144,32,0,6,96
     ,226,2,227,2,0,144
```



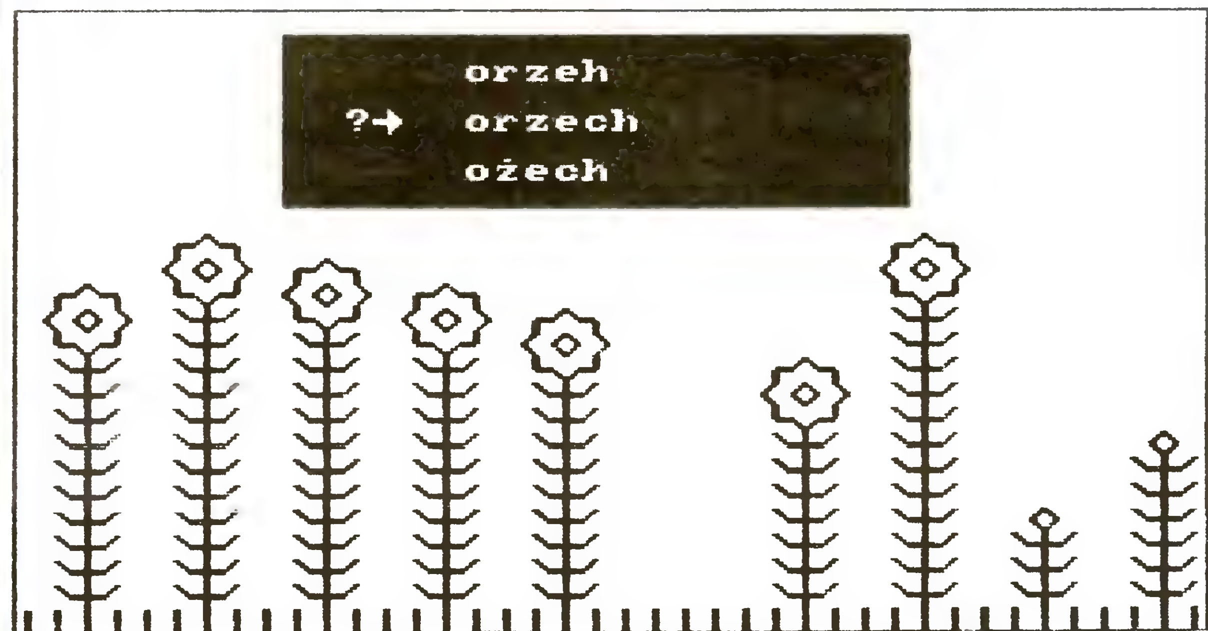
Program „ORTOTEST” opracowany został w języku BASIC na komputerze AMSTRAD/SCHNEIDER CPC 6128. Umożliwia atrakcyjne (w formie testu) sprawdzenie i utrwalenie wiedzy z zakresu ortografii. Użyte w programie wyrazy zostały dopasowane do poziomu II-giej klasy szkoły podstawowej. Czas przeznaczony na odpowiedź oraz ilość i rodzaj używanych w teście wyrazów można łatwo zmienić modyfikując wiersze 480—730. Liczba na końcu wiersza DATA oznacza, który z kolejno przedstawionych wariantów pisowni wyrazu jest poprawny.

Tadeusz Panasiewicz

```
W moim ogrodzie rosna dziwne kwiaty !
Rozkwitaja tylko wtedy, gdy odgadniesz
pisownię wyrazu. Jeśli się pomylisz, to
moga się zapaść pod ziemię ze wstydu.
Nie namyślaj się długo, bo moga zwiędnać
czekajac na Twoja odpowiedź.

Będę kolejno pokazywał różnie napisane
wyrazy. Naciśnij dowolny klawisz, gdy
wskażę wyraz napisany poprawnie.

           Naciśnij klawisz. POWODZENIA !
```




```
Ocena za test: 5

Ilość odpowiedzi złych: 1
Ilość odpowiedzi dobrych: 9

Łaczna długość kwiatków: 98 cm.
```

Ekrany programu ORTOTEST

```
10 '***************************
20 ' ORTOTEST
30 ' T.Panasiewicz, 15.01.90r.
40 '***************************
50 '
60 '******** POLSKIE LITERY ********
70 '
80 '      KLAWIATURA NUMERYCZNA:
90 '    sama        z SHIFT     z CTRL
100 ' s  z. z,      S  Z. Z,     7  8  9
110 ' l  n  o       L  N  O      4  5  6
120 ' a  c  e       A  C  E      1  2  3
130 '      KLAWIATURA LITEROWA:
140 '  a-@  A-!   c-'  C-&   e-[  E-{
150 '  l-^  L-_   n-]  N-}   o-q  O-Q
160 '  s-$  S-#  .z-v .Z-V  ,z-\ ,Z-`
170 SYMBOL AFTER 32
180 SYMBOL 64,0,0,120,12,124,204,118,3          ' a-@
190 SYMBOL 124,24,60,102,102,126,102,102,3      ' A-!
200 SYMBOL 39,24,0,60,102,96,102,60,0           ' c-'
210 SYMBOL 38,8,52,106,192,192,102,60,0         ' C-&
220 SYMBOL 91,0,0,60,102,126,96,60,6            ' e-[
230 SYMBOL 123,254,98,104,120,104,98,254,3      ' E-{
240 SYMBOL 94,56,24,28,24,56,24,60,0            ' l-^
250 SYMBOL 95,240,96,112,96,226,102,254,0       ' L-_
260 SYMBOL 93,24,0,220,102,102,102,102,0        ' n-]
270 SYMBOL 125,24,198,230,246,222,206,198,0     ' N-}
280 SYMBOL 113,24,0,60,102,102,102,60,0         ' o-q
290 SYMBOL 81,12,56,108,198,198,108,56,0        ' O-Q
300 SYMBOL 36,24,0,60,96,60,6,124,0             ' s-$
310 SYMBOL 35,24,60,96,60,6,102,60,0            ' S-#
320 SYMBOL 118,24,0,126,76,24,48,126,0          '.z-v
330 SYMBOL 86,254,198,140,126,48,102,254,0      '.Z-V
340 SYMBOL 92,12,24,126,76,24,48,126,0          ',z-\
350 SYMBOL 96,48,254,172,24,50,102,254,0        ',Z-`
360 KEY DEF 13,1,64,124,49 ' f1
370 KEY DEF 14,1,39,38,50  ' f2
380 KEY DEF 5,1,91,123,51  ' f3
390 KEY DEF 20,1,94,95,52  ' f4
400 KEY DEF 12,1,93,125,53 ' f5
410 KEY DEF 4,1,113,81,54  ' f6
420 KEY DEF 10,1,36,35,55  ' f7
430 KEY DEF 11,1,118,86,56 ' f8
440 KEY DEF 3,1,92,96,57   ' f9
450 '
460 '******** DANE DLA TESTU *********
470 '
480 czas=1000:wyrazy=25:iletesty=10
490 DATA vq^tko,rzq^tko,vu^tko,1
500 DATA chsz@szcz,chrz@szcz,hrz@szcz,2
510 DATA hrzan,chszan,chrzan,3
520 DATA brzqzka,bvuzka,brzuzka,1
530 DATA orzeh,orzech,ovech,2
540 DATA rqrza,ruva,rqva,3
550 DATA kucharz,kuchav,kuharz,1
560 DATA vq^f,vq^w,vu^w,2
570 DATA pszechodzie],przehodzie],przechodzie],3
580 DATA chor@giew,chorongiew,hor@giew,1
590 DATA elementasz,elementarz,elem[tarz,2
600 DATA charcerz,harcev,harcerz,3
610 DATA hu$tawka,hu\dawka,hu$tafka,1
620 DATA kovuh,kovuch,kovqch,2
630 DATA ksionvka,ksi@rzka,ksi@vka,3
640 DATA m[vczyzna,m[vszczyzna,menszczyzna,1
650 DATA korytav,korytarz,korytasz,2
660 DATA narzendzie,nav[dzie,narz[dzie,3
670 DATA nqv,nuv,nqrz,1
680 DATA otrzutowiec,odrzutowiec,odvutowiec,2
690 DATA prqrzniak,pruvniak,prqvniak,3
700 DATA strzykawka,strzykafka,stszykawka,1
710 DATA $rupka,$rubka,$rqbka,2
720 DATA tfarz,twav,twarz,3
730 DATA vaglqwka,vagluwka,rzaglqwka,1
740 '

750 '********* TEST *************
760 '
770 DIM wyrazy$(wyrazy-1,2),kody(wyrazy-1),punkty(wyrazy-1)
780 RESTORE 490
790 FOR n=0 TO wyrazy-1
800   FOR m=0 TO 2:READ wyrazy$(n,m):NEXT m
810   READ kody(n)
820 NEXT n
830 '
840 'instrukcja
850 INK 0,1:INK 1,25:PAPER 0:PEN 1:BORDER 1:MODE 1
860 PRINT"W moim ogrodzie  rosn@  dziwne  kwiaty !"
870 PRINT"Rozkwitaj@  tylko wtedy, gdy  odgadniesz"
880 PRINT"pisowni[  wyrazu. Je$li si[ pomylisz, to"
890 PRINT"mog@ si[ zapa$'  pod  ziemi[ ze  wstydu."
900 PRINT"Nie namy$laj si[ d^ugo, bo mog@ zwi[dn@'"
910 PRINT"czekaj@c na Twoj@ odpowied\.":PRINT:PRINT:PRINT
920 PRINT"B[d[  kolejno pokazywa^ rqvnie  napisane"
930 PRINT"wyrazy.  Naci$nij  dowolny  klawisz, gdy"
940 PRINT"wskav[ wyraz napisany poprawnie.":PRINT:PRINT:PRINT
950 PRINT"             Naci$nij klawisz. POWODZENIA !"
960 CLEAR INPUT
970 IF INKEY$="" THEN 970
980 '
990 'rysunek
1000 CLS:BORDER 14:INK 0,22:INK 1,9:INK 2,6
1010 FOR x=1 TO 40:LOCATE x,25:PRINT CHR$(149);:NEXT x 'trawa
1020 FOR x=2 TO 40 STEP 4 'kwiatki
1030   FOR y=24 TO 19 STEP -1:GOSUB 1100:NEXT y
1040   y=19:GOSUB 1120
1050 NEXT x
1060 GOTO 1220
1070 '
1080 'elementy rysunku
1090 '
1100 'łodyga
1110 LOCATE x,y:PRINT CHR$(193);CHR$(159);CHR$(192);:RETURN
1120 'zarodek
1130 PEN 2:LOCATE x,y-2:PRINT CHR$(32);CHR$(32);CHR$(32);
1140 LOCATE x,y-1:PRINT CHR$(32);CHR$(202);CHR$(32);:PEN 1
1150 RETURN
1160 'kwiat
1170 PEN 2:LOCATE x,y-2:PRINT CHR$(150);CHR$(196);CHR$(156);
1180 LOCATE x,y-1:PRINT CHR$(199);CHR$(202);CHR$(197);
1190 LOCATE x,y-0:PRINT CHR$(147);CHR$(198);CHR$(153);:PEN 1
1200 RETURN
1210 '
1220 WINDOW #1,10,30,2,8:PAPER #1,2:PEN #1,0:CLS#1
1230 ENV 1,5,1,1,5,0,1,15,-1,6:ENT -1,10,1,1,20,-1,1,10,1,1
1240 ENV 2,5,1,1,4,0,5,15,-1,5:bu=900
1250 ten$="?"+CHR$(243)
1260 '
1270 'test
1280 FOR n=0 TO iletesty-1 'tyle razy pytam
1290   DIM blad(2):x=n*4+2 'lokacja kwiatka
1300   numer=INT(RND*wyrazy) 'losowanie
1310   IF kody(numer)<1 THEN 1300 'czy wydano ten numer
1320   CLEAR INPUT:CLS#1
1330   FOR m=0 TO 2 'druk wyrazu
1340     LOCATE #1,7,m*2+2:PRINT#1,wyrazy$(numer,m)
1350   NEXT m
1360   dobry=kody(numer)
1370   FOR y=20 TO 25 'czas odpowiedzi
1380     FOR m=1 TO 3 'wskazanie wyrazu
1390       IF blad(m-1)>0 THEN 1450
1400       LOCATE#1,3,m*2:PRINT#1,ten$
1410       SOUND 1,y*10+m*200,100,10,1,1
1420       FOR t=0 TO czas
1430         IF INKEY$<>"" THEN 1740 'tam sprawdzanie
1440       NEXT t
1450       LOCATE #1,3,m*2:PRINT#1,"  "
1460     NEXT m
1470     GOSUB 1120 'cVut marnieje
1480   NEXT y
1490   GOSUB 2030 'zostanie tylko dobry wyraz
1500   GOSUB 1980:LOCATE x,24:PRINT"   " 'nie ma kwiatka
1510   y=25:GOSUB 1980 'kara
1520   punkty(n)=25-y
1530   kody(numer)=0:ERASE blad
1540 NEXT n
1550 '
1560 'koniec testu
1570 dobry=0:wynik=0
1580 FOR n=0 TO iletesty-1
1590   IF punkty(n)>0 THEN dobry=dobry+1
1600   wynik=wynik+punkty(n)
1610 NEXT n
1620 BORDER 1:INK 0,1:INK 1,24:PAPER 0:PEN 1:CLS
1630 LOCATE 1,6:PRINT"Ocena za test:";2+INT(dobry/3)
1640 LOCATE 1,10
1650 PRINT"Ilo$' odpowiedzi z^ych:";iletesty-dobry
1660 LOCATE 1,12:PRINT"Ilo$' odpowiedzi dobrych:";dobry
1670 LOCATE 1,16:PRINT"_@czna d^ugo$' kwiatkqw:";wynik;"cm."
1680 FOR n=0 TO 1
1690   FOR m=1 TO 2:GOSUB 2100:NEXT m
1700   SOUND 1,142,100:FOR t=0 TO 3000:NEXT t
1710 NEXT n
1720 RUN
1730 '
1740 'dobrze czy niedobrze
1750 LOCATE#1,3,m*2:PRINT#1,"  ":SOUND 1+128,0
1760 IF dobry=m THEN 1770 ELSE 1850
1770 'dobrze
1780 GOSUB 2030 'zostanie dobry wyraz
1790 z=y-1:FOR y=z TO z-6 STEP -1 'kwitnienie
1800   GOSUB 1100:GOSUB 1120
1810 NEXT y:GOSUB 1160
1820 GOSUB 2100:SOUND 1,142,100 'nagroda
1830 FOR t=0 TO 3000:NEXT t
1840 GOTO 1520
1850 'niedobrze
1860 blad(m-1)=1
1870 LOCATE #1,7,m*2
1880 PRINT#1,STRING$(LEN(wyrazy$(numer,m-1)),"-")
1890 IF blad(0)+blad(1)+blad(2)>1 THEN 1950 'dwa byki=koniec
1900 FOR z=1 TO 2
1910   GOSUB 1120 'kwiatek marnieje
1920   IF y=25 THEN 1490 ELSE y=y+1
1930 NEXT z
1940 y=y-1:GOSUB 1980:GOTO 1480
1950 '2*niedobrze
1960 z=y:FOR y=z TO 25:GOSUB 1120:NEXT y:GOTO 1500
1970 '
1980 'kara
1990 SOUND 1,bu,100,10,2:SOUND 1,bu,100,10,2
2000 SOUND 1,bu,25,10,2:SOUND 1,bu,75,10,2
2010 FOR m=0 TO 3500:NEXT m:RETURN
2020 '
2030 'wymazanie niedobrych wersji
2040 FOR m=0 TO 2
2050   LOCATE #1,7,m*2+2
2060   k$=wyrazy$(numer,m)
2070   IF m+1<>dobry THEN PRINT#1,STRING$(LEN(k$),"-")
2080 NEXT m:RETURN
2090 '
2100 'nagroda
2110 DATA 95,95,142,127,119,106,95,95,119,95,95,119,95
2120 DATA 95,142,119,142,179,119,142,142,106,119,127,-1
2130 RESTORE 2110:READ nuta
2140 WHILE nuta<>-1:SOUND 1,nuta,35,12:READ nuta:WEND
2150 RETURN
```

Listing programu ORTOTEST

OPERACJE DYSKOWE W SYSTEMIE CP/M Plus

Cz. IV — odczyt dyskietek ZX Spectrum (FDD 3000)

W tej części zajmiemy się logiczną organizacją dyskietki, tzn. rozmieszczeniem plików i strukturą katalogu. Efektem praktycznym będzie prosty program pozwalający na odczyt plików z dyskietek ZX Spectrum na komputerze Amstrad.

Logiczna organizacja dyskietki

Podstawową jednostką informacji z punktu widzenia modułu BDOS (ang. Basic Disk Operating System) jest rekord logiczny zawierający 128 bajtów. Zapis pliku na dysku polega na zapisie kolejnych rekordów, z których plik się składa. Oprócz tego na dysku zapisywana jest informacja o pliku, a więc jego nazwa i położenie na dysku.

Pamiętanie pozycji każdego rekordu zajmowałoby dużo miejsca, ograniczając obszar przeznaczony na dane, tzn. pliki. Z tego powodu najmniejszą porcją informacji opisywanej przez system na dysku jest tzw. jednostka alokacji pamięci dyskowej, grupująca 8 lub więcej rekordów. Jeśli jednostek tych, zwanych również blokami alokacji, jest mniej niż 256, to na ponumerowanie każdej z nich potrzebujemy jeden bajt. Pierwszych kilka bloków (maksymalnie 16, a zwykle 2) przeznaczonych jest na katalog dysku. Pozostałe bloki są dynamicznie przydzielane poszczególnym plikom na dysku.

Struktura katalogu — tablica FCB

Podstawowym elementem katalogu jest 32-bajtowa tablica FCB (ang. File Control Block). W jednym rekordzie mieści się 4, a w dwóch jednokilobajtowych blokach (obszar zwykle rezerwowany na katalog) — 64.

Każda tablica FCB, nazywana także pozycją katalogu (ang. directory entry), pozwala opisać jeden plik, jeśli jego wielkość nie przekracza pewnej liczby rekordów. Dłuższe pliki dzielone są na segmenty (ang. file extents) i potrzebują na opis więcej niż jedną pozycję. Struktura tablicy FCB została przedstawiona w Tab. 1.

Pierwszych dwanaście bajtów (0—11) tablicy FCB służy do przechowywania numeru użytkownika i nazwy pliku łącznie z rozszerzeniem. Bajty o numerach 12, 14 i 15 określają rozmiar pliku w sposób przedstawiony na rys. 1.

Na zapamiętanie numerów bloków alokacji, przydzielonych plikowi, przeznaczono 16 ostatnich bajtów każdej pozycji katalogu. Jeśli bloków jest więcej niż 256 to ich numerowanie wymaga dwóch bajtów na blok i każda pozycja katalogu może rezerwować tylko 8 bloków. W przypadku mniejszej ich liczby każda pozycja pozwala przyporządkować 16 bloków.

W systemie CP/M Plus wprowadzono także inne tablice FCB realizujące system haseł, etykietę dysku i datowanie zbiorów. Rozszerzenia te nie będą tu opisane.

Dyskietka ZX Spectrum

Komputer ZX Spectrum z podłączoną stacją dysków FDD 3000 f-my Timex umożliwia pracę w dwóch systemach operacyjnych: TOS i CP/M 2.2. Sposób przechowywania zbiorów na dyskietce i jej struktura fizyczna i logiczna w obu systemach jest prawie taka sama. Dyskietka jest formatowana jednostronnie i zawiera 40 ścieżek. Na każdej ścieżce znajduje się 16 sektorów o długości 256 bajtów każdy i numerowanych od 0 do 15. Jednostka alokacji pamięci na rozmiar 1 KB i zawiera cztery sektory.

W Tab. 2 podano numerację bloków alokacji dla TOS'u i dwóch wersji systemu CP/M — Timex i Polbrit. Na dyskietce TOS'u cztery pierwsze ścieżki zarezerwowane są na system, a katalog zawierający 128 pozycji zajmuje całą piątą ścieżkę (bloki o numerach 0, 1, 2, 3). Na pliki przeznaczone są ścieżki od szóstej do czterdziestej (bloki 4, 5...143), czyli 140 KB.

Dyskietka CP/M'u (Timex) zawiera dwie ścieżki systemowe, katalog — 128 pozycji — znajduje się na trzeciej ścieżce, a na pliki, zajmujące bloki 4,5..151, przeznaczono 148 KB. Wersja Polbritu ma mniejszy katalog — 64 pozycje — i o dwa bloki więcej na pliki.

Opisana powyżej struktura dyskietki ZX Spectrum pozwala na zdefiniowanie tablic DPB potrzebnych na komputerze Amstrad do pracy z dyskietkami TOS'u i CP/M'u (Timex i Polbrit). Tablice te zebrano w TAB. 3, gwiazdkami zaznaczając pola zależne od systemu.

Przeplot

W systemie CP/M Plus komputerów Amstrad CP/M nie jest używana tablica przeplotów. Z kolei zarówno TOS, jak i CP/M (Timex i Polbrit) mają taki przeplot, tzn. numery sektorów fizycznych różnią się do numerów sektorów logicznych. Dla TOS'u przeplot wynosi 7, a dla obu CP/M'ów ZX Spectrum — 5. W TAB. 4 zebrano tablice „tłumaczące" przeplot dla obu systemów.

Program ZxRead

Dysponując informacjami przedstawionymi w tym artykule i biblioteką procedur DISK3.SYS, opisaną w części II (Bajtek 3—4/90) możemy napisać stosunkowo krótki i prosty program, który pozwoli nam na odczyt dyskietek z komputera ZX Spectrum na Amstradzie (listing 1.).

W programie znajdują się trzy procedury i jedna funkcja. Ta ostatnia ma znaczenie pomocnicze i zamienia liczbę całkowitą na ciąg znaków w systemie szesnastkowym. Pierwsza z procedur — DefineDPB — definiuje odpowiednią tablicę DPB (na listingu — TOS). Procedura ShowDMAbuffer wyświetla zawartość bufora (w programie jest to 256-bajtowy sektor). Ostatnia z procedur — ReadABlock — korzystając ze zdefiniowanej w pliku DISK3.SYS procedury niskiego poziomu DD_Read, czyta blok alokacji z uwzględnieniem przeplotu.

Program przedstawiony na listingu pozwala na pracę z dyskietkami TOS'u. Jeśli chcemy korzystać z dyskietek CP/M'u (Timex lub Polbrit) musimy zmienić zawartość procedury DefineDPB i zmienną XLT, definiującą przeplot. Oprócz tego, jeśli posługujemy się dyskietkami Polbritu, to konieczna jest zmiana w jednej z instrukcji w procedurze ReadABlock:

track := BlockNo div 4+2;

Program działa na komputerze wyposażonym w dwa napędy dysków (dla PCW drugim napędem jest Ramdysk M:). Do napędu A wkładamy dyskietkę TOS'u, a program uruchamiamy z innego napędu. Możliwe jest „ręczne" skopiowanie jednego pliku z dyskietki ZX Spectrum na dyskietkę Amstrada.

Na początku wybieramy sektor katalogu, który może zawierać interesujące nas zbiory. Należy podać numer fizyczny sektora (0, 7, 14 ....) i numer ścieżki 4 (dla CP/M'u — 2). Wprowadzając −1 na pytanie o numer sektora kończymy ich oglądanie i przystępujemy do kopiowania wybranego pliku. Należy odpowiadając na kolejne pytania podać nazwę zbioru, pierwszy i ostatni zajmowany przez niego blok alokacji. Działanie programu zilustrowano na rys. 2.

Opisywany program mimo modelowego charakteru jest bardzo użyteczny i razem z innymi wersjami (dla CP/M'u) był wielokrotnie używany. Możliwe są jego modyfikacje mające na celu zwiększenie komfortu pracy. Dotyczy to zwłaszcza automatycznego znajdowania bloków alokacji danego pliku, pracy z jednym napędem i kopiowania więcej niż jednego zbioru. W podobny sposób, wykorzystując procedurę DD_Write można zapisywać pliki „amstradowe" na dyskietkach ZX Spectrum. Ponadto, jeśli zastąpimy procedury niskiego poziomu charakterystyczne dla komputerów firmy Amstrad innymi, to możliwe jest napisanie podobnych programów na inne komputery np. IBM PC.

Jonasz Mayer

Tab. 1. Tablica FCB

bajt	nazwa	znaczenie
0	user	numer użytkownika (0-15)
1-8	name	nazwa pliku (8 znaków)
9-11	ext	rozszerzenie (typ) pliku (3 znaki)
12	ex	młodsze bity (0-4) numeru segmentu (ang. file extent)
13	s1	nieużywane
14	s2	starsze bity (5-10) numeru segmentu
15	rc	liczba rekordów w segmencie
16-31	d0-dN	numery przydzielonych segmentowi bloków alokacji

```
{****************************************************************}
TAB. 2. Numeracja blokow alokacji dyskietki: (a) - TOS,
    (b) - CP/M (Timex), (c) - CP/M (Polbrit).
{****************************************************************}

 Sciezka           Bloki alokacji              (a)
 (Track)    A      B        C        D

   00              system TOS
   01              system TOS
   02              system TOS
   03              system TOS

   04       0      1        2        3         (Katalog)

   05       4      5        6        7         (Pliki)
   06       8      9        10       11        (Pliki)
   :.........:........
   39       140    141      142      143       (Pliki)

{****************************************************************}

 Sciezka           Bloki alokacji              (b)
 (Track)    A      B        C        D

   00              system CP/M  (TIMEX)
   01              system CP/M  (TIMEX)

   02       0      1        2        3         (Katalog)

   03       4      5        6        7         (Pliki)
   04       8      9        10       11        (Pliki)
   :.........:........
   39       148    149      150      151       (Pliki)

{****************************************************************}

 Sciezka           Bloki alokacji              (c)
 (Track)    A      B        C        D

   00              system CP/M  (POLBRIT)
   01              system CP/M  (POLBRIT)

   02       0(K)   1(K)     2(P)     3(P)  (Katalog+Pliki)

   03       4      5        6        7         (Pliki)
   04       8      9        10       11        (Pliki)
   :.........:........
   39       148    149      150      151       (Pliki)

{****************************************************************}
```

```
S2 S2 S2 S2 S2 S2  EX EX EX EX EX
 5  4  3  2  1  0   4  3  2  1  0
                                  RC RC RC RC RC RC RC RC
 +                                 7  6  5  4  3  2  1  0
---------------------------------------------------------
18 17 16 15 14 13 12  11 10  9  8  7  6  5  4  3  2  1  0
```

Rys. 1. Tworzenie 19-bitowej liczby rekordów pliku na podstawie pól EX, S2 i RC.

```
ZX Spectrum in A:
  32   0   3   7   0 143   0 127   0 240   0  32   0   4   0   1   1   0  40
sector: 0  7
track: 4

0000   02 43 41 4C 45 4E 44 41 52 C2 41 53 00 7A 00 18    .CALENDAR.AS.z..
0010   09 0A 0B 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
0020   02 43 4F 4C 4F 55 52 53 20 A0 20 20 00 8D 00 02    .COLOURS .  ....
0030   0C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
0040   02 47 41 4D 45 53 20 20 20 C4 49 52 00 00 00 00    .GAMES   .IR....
0050   04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................

0060   04 44 49 43 45 20 20 20 20 C2 41 53 00 E3 00 14    .DICE    .AS....
0070   26 27 28 00 00 00 00 00 00 00 00 00 00 00 00 00    &'(.............

0080   04 48 49 2D 4C 4F 20 20 20 C2 41 53 00 11 00 0A    .HI-LO   .AS....
0090   10 11 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00A0   03 54 59 50 45 20 20 20 20 A0 20 20 00 23 00 04    .TYPE    .  .#..
00B0   0D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00C0   00 46 49 4C 49 4E 47 20 20 C4 49 52 00 00 00 00    .FILING  .IR....
00D0   05 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00E0   05 41 44 44 52 45 53 53 20 C2 41 53 00 2E 00 0A    .ADDRESS .AS....
00F0   14 15 00 00 00 00 00 00 00 00 00 00 06 00 00 00    ................

sector: 7  -1
File name: DICE.BAS
First Block No: $26
Last  Block No: $28
```

Rys. 2. Przykladowe wykonanie programu ZXread.

TAB. 3. Tablice DPB dla TOS'u i CP/M'u

	TOS	CP/M (T)	CP/M (P)	
LSPT	32	32	32	
BSH	3	3	3	
BLM	7	7	7	
EXM	0	0	0	
DSM	143	151	151	*
DRM	128	128	64	*
ALO	F0h	F0h	C0h	*
CKS	20h	20h	10h	*
OFF	4	4	2	
PSH	1	1	1	
PHM	1	1	1	
Sideness	0	0	0	
TPS	40	40	40	
PSPT	16	16	16	
FSN	0	0	0	
SS	100h	100h	100h	
RWGAP	16h	16h	16h	
FGAP	2Ah	2Ah	2Ah	
MFM	60h	60h	60h	
AUTO	FFh	FFh	FFh	

TAB. 4. Tablice przeplotów dla systemu CP/M i TOS. Nr jest numerem logicznym sektora, a wierszach TOS i CP/M podano odpowiednie numery sektorow fizycznych

Blok	A				B				C				D			
Nr	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CP/M	0	5	10	15	4	9	14	3	8	13	2	7	12	1	6	11
TOS	0	7	14	5	12	3	10	1	8	15	6	13	4	11	2	9

LISTING 1

```
Program ZXread;  { ZX spectrum }
{***************************************************************}
{           Plik ZXR.PAS   (C) JM Nov 24,89                      }
{***************************************************************}

{$I DISK3.SYS }    { dolaczenie procedur dyskowych  }
                   { niskiego poziomu               }

Procedure DefineDPB (var DPB : DPBrec);
{***************************************************************}
{ Definiuje tablice DPB opisana w ponizszych komentarzach.      }
{***************************************************************}
begin
  with DPB
  do begin
       LSPT := 32;                  { 32 sektory  log. na sciezke }
       BSH  := $03;  BLM := $07;           { Blok alokacji 1KB }
       EXM  := $00;                         { 16 KB na segment }
       DSM  := 143;                        { 144 bloki  (144 KB) }
       DRM  := $7F;                     { 128 pozycji katalogu }
       AL0  := $F0;   { 4 pierwsze bloki alokacji na katalog }
       CKS  := $20;      { Wielkosc sumy kontrolnej katalogu }
       OFF  := $04;                 { 4 zarezerwowane sciezki }
       PSH  := $01;  PHM := $01;    { sektor fiz. 256 bajtow }
       Sideness := $00;                   { 1 - stronny naped }
       TPS  := $28;                     { 40 sciezek na strone }
       PSPT := $10;       { 16 sektorow fizycznych na sciezke }
       FSN  := $00;               { numer pierwszego sektora }
       SS   := $0100;           { sektor fizyczny 256 bajtow }
       RWGAP:= $16;  FGAP:= $2A;       { przerwy r/w i format }
       MFM  := $60;                      { zapis w trybie MFM }
       AUTO := $ff;               { bez autodetekcji formatu }
     end;
end; { of Define DPB }

{***************************************************************}
var
   ch, drive    : char;                     i : byte;
   SDS          : SDSrec;   sec,trk : integer;
   DPB, OldDPB  : DPBrec;
type
    hexstr=string[4];
{***************************************************************}

function Hex (Number: Integer;  Bytes: Integer) : HexStr;
{***************************************************************}
{ Zamienia liczbe dziesietna na lancuch znakowy w systemie       }
{ szesnastkowym.                                                 }
{***************************************************************}
const
     T : array(.0..15.) of Char = '0123456789ABCDEF';
var
     D : Integer;      H : HexStr;
begin
   H(.0.) := Chr (Bytes+Bytes);
   for D := Bytes+Bytes downto 1
   do begin
          H(.D.) := T(.Number and 15.);
          Number := Number shr 4
      end;
   Hex := H
end; (* of proc Hex *)

procedure ShowDMAbuffer (adr: integer; max : byte);
{***************************************************************}
{ Wyswietla zawartosc bufora.                                    }
{***************************************************************}
var   b : byte;
      address, i,j  : integer;   ch : char;
begin
  clrscr;
  address := 0;
  write(#27#48); { status line off }
  for i:=0 to max
  do begin   writeln;
       write( Hex(address,2),'  ');
       for j := 0 to 15
       do write (Hex (mem[adr+i*16+j],1):3);
       write('     ');
       for j := 0 to 15
       do begin
            b := mem[adr+16*i+j];
            if b in [32..126]
            then write (Chr (b))
            else write ('.');
          end;
       address := address + 16;
     end;
  repeat until KeyPressed;                    read (kbd,ch);
  writeln (#27#49); { status line on  }
end; { of ShowDMAbuffer }

{***************************************************************}
const
    XLT : array [0..15] of byte =
          (0,7,14,5, 12,3,10,1, 8,15,6,13, 4,11,2,9);
type
    BlockType = array [0..1023] of byte;
var
    Block : BlockType;    name            : string [20];
    f     : file;         BlockNo, LB,FB  : integer;
{***************************************************************}

procedure ReadABlock (BlockNo : integer; Var Block : BlockType);
{***************************************************************}
{ Porcedura czyta blok alokacji z uwzglednieniem przeplotu      }
{***************************************************************}
var
   i,track,FSNLB : integer;
begin
  track := BlockNo div 4 + 4;
  FSNLB := (BlockNo mod 4) * 4;
  for i := 0 to 3
  do begin
       DD_Read ('A',track,XLT [FSNLB+i]);
       Move (Buf, Block [i*256],256);
     end;
end; (* read allocation Block *)

{***************************************************************}

begin  { MAIN }
  ClrScr;     writeln ('ZX Spectrum in A:');

  drive := 'A';                 { wybor napedu 3" }
  GetDPB (drive,OldDPB);   { zapamietanie poprzedniego DPB    }

  DefineDPB (DPB);
  SetDPB (drive,DPB);                          { Zadanie formatu }
  for i:= 0 to 16 do write (Mem[i+$FF4a]:4);
  for i:=17 to 26 do write (Mem[i+$FF4a]:4);

  Sec := 0;  Trk := 4;

  repeat
    writeln;
    write ('sector: ',sec,'  ');  readln (sec);
    if sec=-1 then else begin
    write ('track: ',trk,'  ');   readln (trk);
    DD_Read (drive,trk,sec);
    ShowDMAbuffer (addr(buf),15);
                           end;
  until sec=-1;

  write ('File name: '); readln (name);
  assign (f,name);
  rewrite (f);

    write ('First Block No: '); readln (FB);
    write ('Last  Block No: '); readln (LB);

    for BlockNo := FB to LB
    do   begin
           ReadABlock (BlockNo, Block);
           BlockWrite (f,Block,8);
         end;

  close (f);

  SetDPB (drive,OldDPB);      { odtworzenie poprzedniego DPB }
end.

{***************************************************************}
```



POZIOM

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

START

1.

2.

3.

I

II

III





UNDER-WURLDE

□□ - ROŚLINY, ΔΔ - KRATERY,
⊗ - KAMIEŃ, I; II; III - INSEKTY,
⇧ - WYJŚCIE, **1.2.3.** - BROŃ DO ZABICIA INSEKTÓW, (I-1; II-2; III-3)

Rys. Zygmunt Zaradkiewicz

27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52



10 BAJTKOWA LISTA PRZEBOJÓW 7–8

Ostatnio nie notuje się zalewu rynku grami. Jest to zapewne wynikiem sezonu urlopowo-wakacyjnego. Mniejsze jest zainteresowanie wszystkim, a więc i ukochanym komputerkiem.

Na pierwszym miejscu Listy — Aliens. Ten renesans spowodowało zapewne ukazanie się wersji tej gry na Spectrum, przetrzymanej kilka lat przez naszych handlarzy. Przypominamy: opis i mapa były w „zerowym" numerze „Bajtka" — „Top Secret", który usamodzielnił się.

Liczymy na erupcję naprodukowanych przez wakacje gier. Prawdopodobnie będą one nadal pokątnie sprowadzane do Polski. Jak długo?

		ATARI	AMSTRAD	COMMODORE	SPECTRUM
1	Aliens			×	<
2	Iron Lord	>		×	<
3	The Untouchables	>	<	×	×
4	Batman	>	<	×	×
5	Test Drive II	>	×	×	×
6	Cabal	>	>	×	<
7	Fighter Bomber	>		×	<
8	Robbo	<			
9	Total Eclipse		<	×	<
10	Red Heat	>	<	<	<



Kopertę nadesłał Mateusz Stanisz z Lwówka Śl.



UNDERWURLDE

Bard spojrzał na wypchany mieszek z pieniędzmi — dorobek całego życia. Zawód Barda był łatwy, lecz bardzo niebezpieczny. Zajmował się ratowaniem księżniczek i zabijaniem smoków. Dziś miał zamiar udać się do zamku króla Wacława II na turniej rycerski. Nagrodą było sto sztuk złota oraz królewna Izabela zwana Piękną. Bard schował mieszek do komody, wziął kopię, założył zbroję i wyszedł z chaty. Prawie bez sensacji wsiadł na konia, jeśli nie liczyć złamanego nosa wieśniaka, który podszedł za blisko kopii.

Jadąc do zamku kierował się znakami umieszczonymi przy drodze. Po kilku dniach zdziwiony zauważył, że nie widzi swoich konkurentów podążających na turniej. Już zamierzał zrezygnować z dalszej jazdy, gdy zobaczył wysokie wieże zamku. Wszyscy powitali go serdecznie i wylewnie. Okazało się, że konkurs został odwołany, ponieważ królewna ciężko zachorowała. Zatruła się nieznanym grzybem znalezionym w lesie. Uratować mógł ją tylko wywar z cudownych kwiatów i, jak się domyślacie, rośliny te rosły tylko w jednym miejscu na świecie. Bard nie zastanawiał się długo, postanowił udać się po ten cudowny lek.

Kwiaty te rosły w podziemniach zamku Johna Czekoladowe Oko. Był to człowiek okrutny i ponury, oprócz tego zabijał każdego, kto nie przyniósł mu czekolady. Nasz bohater nie był głupi i zaopatrzył się w sporą ilość tego artykułu. Teraz już mógł bezpiecznie udać się do zamku Johna. Ten jednak był w bardzo złym nastroju i powiedział, że daje Ci 10 godzin na znalezienie rośliny. Od razu udałeś się do podziemi i rozpocząłeś poszukiwania. W tym momencie Ty wgrałeś tę grę i sterujesz Bardem. Pamiętaj, że tylko od Ciebie zależy jego życie, a więc i księżniczki.

Poruszasz się w bardzo skomplikowanym labiryncie, z którego są trzy wyjścia. Aby się do nich dostać musisz zabić trzy potwory blokujące drogę. Każdego z nich unicestwi inna broń. W pierwszej komnacie znajdziesz kamień, którym będziesz mógł odpędzać od siebie natrętne stworki przeszkadzające w poruszaniu się. Czasami przemieszczanie się jest możliwe tylko za pomocą baniek wypuszczanych przez wulkany. Powrót ułatwi Ci lina, trzeba się jej jednak dobrze złapać.

Każdy upadek z większej wysokości kończy się śmiercią. Czasami można znaleźć maść, dzięki której spadek jest niegroźny, działa ona jednak tylko pół minuty. Upadek neutralizuje także bańka z wulkanu, trzeba na nią trafić. Po wyjściu z labiryntu czeka Cię wspaniała niespodzianka. Dowiadujesz się, że poszukiwania musisz kontynuować w jednej z popularnych gier, np.: KNIGHT LORE, PENTAGRAM itp. Każdy gracz — twardziel wytrzyma ten cios i wgra potrzebny program. Inni dadzą spokój i położą się do łóżka na upragniony odpoczynek. Każdy jednak zorientuje się, że końca grania nie widać.

Firma: Ultimate
Komputer: Spectrum, Commodore, Amstrad

Luke



COMBAT SCHOOL

Jasnożółta tarcza słońca powoli, z majestatem wschodziła na horyzoncie. Wśród długich cieni i kumkania żab na pobliskim bagnie, do życia budził się niewielki wojskowy obóz. Już wkrótce rozlegnie się sygnał dzwonka i śpiący do tej pory wartownicy wyrwą się z długiego jak na żołnierskie warunki, sześciogodzinnego letargu. Zaraz po nich ze snu zerwą się i inni żołnierze rozpoczynając nowy dzień służby.

Długie stojące na palach baraki wydawały się być młodymi stonogami. Poprzewieszane między nimi sznurki na bieliznę kołysały się, podrzucane nielicznymi powiewami wiatru. Pod nimi leżały mokre jeszcze skarpetki, zbyt słabo umocowane przez swych właścicieli. Mimo prania, zapach jakie wydzielały byłby śmiertelny dla każdego niezaprawionego i nieprzygotowanego. Być może stały się one przyczyną śmierci pokiereszowanej żaby, której zwisająca ze schodów głowa przypominała łeb czatującego węża.

Nagle drzwi baraku otworzyły się i wyjrzała z nich prawie całkowicie łysa głowa żołnierza. Oprócz włosów widać było już tylko jego nos, przypominający wygięty, spłaszczony na końcu klucz francuski. Nawet najbardziej niedomyślny obserwator zorientowałby się, że ani to wygięcie ani spłaszczenie nie mogło pochodzić od rodziców. Każdy wiedział doskonale, że pewnego dnia, człowiek teh był na tyle nieostrożny lub natrętny, że wpadł na czyjąś pięść, szczęśliwie tracąc od razu przytomność.

Żołnierz widząc śpiącego wartownika rozochocił się wyraźnie, szerzej otoworzył drzwi i wyszedł na zewnątrz. Ponieważ nie miał wzrostu koszykarza, nie musiał schylać się pod niskim daszkiem. Zbliżył się do schodów i zamierzał właśnie szybkimi krokami je pokonać, gdy nadepnął na żabę. Nogi momentalnie wystrzeliły mu do przodu, ciągnąc całe ciało w dół, po schodach. Głowa miarowo odbijała się, na każdym stopniu wydając dźwięk o innej wysokości, tworząc istną Arię Pogrzebową. Ślina bulgotała mu między zębami a szczęki wygrywały rytm pieśni grabarzy. Ręce i tułów, niepowstrzymywane siłą mięśni ruszały się zupełnie swobodnie. Na dole żołnierz wydawał się jeszcze bardziej łysy i bardziej chory niż przedtem. Nawet rodzona matka nie uwierzyłaby, że ta bezładna masa była kiedyś jej synem.

* * *

— „To wasza robota, wy małe, świńskie karaluchy" — krzyczał na całe gardło major Jacky.

Powodem jego zdenerwowania był połamany nieszczęśliwiec, który spadł z samego rana ze schodów tracąc sztuczną szczękę i po raz piąty łamiąc nos. Major Jacky uważał, że winę za to ponoszą żołnierze i nie mógł się powstrzymać, by nie wyładować na nich swojej złości. Nienawidził bowiem wszelkich przejawów świńskich dowcipów w obozie i nie tylko. Cechę tę nabył za czasów swojej służby, kiedy koledzy z drużyny podrzucili mu butelkę z klejem na miejsce szamponu. Klej był na tyle mocny, iż ówczesny jeszcze kapral musiał wziąć kąpiel w rozpuszczalniku. Nawet dzisiaj, po tysiąckrotnym perfumowaniu, strzyżeniu i pielęgnowaniu skóry, jego głowa wydzielała zapach czyściutkiej benzyny. Dowcipni mówili często, że „pan major" przed chwilą wrócił z rafinerii. Szeptano to jednak tylko po cichu i w małym gronie, gdyż „pan major" znany był z sadyzmu. I właśnie dzisiejszy nieszczęśliwy wypadek zamierzał wykorzystać, aby wyładować swoją energię na tym razem niewinnych żołnierzach.

Łysawy major był tego dnia bardzo pomysłowy i wykorzystał wszelkie dostępne środki, aby zgnoić żołnierzy. Najpierw bieg z przeszkodami na czas, a po drodze aż roi się od rowów, murów i drabinek. Potem karabin maszynowy do ręki i biegiem na strzelnicę. Tam czeka oczywiście niespodzianka — tarcze chowają się i pojawiają w bardzo krótkich odstępach czasu.

Trzecią dyscypliną zadaną przez majora był bieg ‚tylko dla twardzieli". Nie każdy potrafi bowiem biec w sprinterskim tempie przez kamienie pomieszane z błotem, aby następnie przesiąść się do łódki i przebyć dość szeroką i zatrzcinioną rzekę.

Kolejne konkurencje miały sprawdzać głównie dobre oko żołnierzy. Różne karabiny i amunicja oraz wymyślne cele, to było naprawdę dobre ćwiczenie. Gdy uda Ci się je zaliczyć, pora sprawdzić samego Szefa. Major czeka na Ciebie z otwartą szczęką i zamkniętymi pięściami. Jeżeli nie zgasisz go na kilka miesięcy wymyśli dla Twoich kumpli takie ćwiczenia, że zginą w męczarniach, porównywalnych tylko do gry w Decathlon. Życzę więc powodzenia, i zapewniam, że ukończenie Combat School jest w pełni możliwe.

Firma: Ocean
Komputer: Spectrum, Commodore, Amstrad

LUKE

S.O.S.

Poszukuję gier WHO DARES WINS II, UNIVERSAL HERO, WORLD KARATE CHAMPIONSHIP. W zamian za gry takie jak ROAD RACE, POLE POSITION i inne. **Łukasz Nowak, ul. Z. Czarnego 48 44–200 Rybnik**

W zamian za opisy do ACE OF ACES, SPELLBOUND, SPINDIZZI i UNIVERSAL HERO oferuję około 150 gier i opisów. **Paweł Noworyta, 32–555 Zagórze 670**

Nie wiem jak posługiwać się programem RAMDISK. Poszukuję opisu oraz nieśmiertelności do gry SUPERMAN. W zamian odstąpię opisy do MONTEZUMA's REVENGE, ROAD RACE, BRUCE LEE i THE LAST STARFIGHTER. Wszystko na Atari 65XE, **Daniel Kownacki, ul. II Pułku Ułanów 30c m 1 72–320 Trzebiatów**

Proszę o przysłanie mi dowolnego programu graficznego na Atari 65XE. **Adam Bukowiński, ul. Ziemowita 7/3 44–100 Gliwice woj. katowickie**

Poszukuję opisów do gier GAUNTLET, SUN DOG, MERCENARY I, II, 10th FRAME WIZBALL i KING's QUEST I. Posiadam Atari 520ST i 50 gier. **Mikołaj Lewkow, ul. Reja 3a 83–300 Kartuzy woj. gdańskie**

Posiadam Commodore 16. W zamian za gry NOSFERATU, POP EYE, SABOTEUR II, TRASHMAN oddam inne gry na ten komputer. **Adam Doliwka, ul. Mokra 11/21 95–200 Pabianice**

Mam kłopoty z grą ROBO KNIGHT. Po uruchomieniu gry muszę co chwila przełączać porty. Proszę także o dokładne opisy i nieśmiertelności do gier FINDERS KEEPERS, AUF WIEDERSEHEN MONTY, HEEBIE JEEBIE i GHOST TOWN. **Jacek Tomala, ul. Szkolna 10 44–244 Żory-Osiny**

W zamian za opisy do gier ZORRO, SABOTEUR II, ALIENS oferuję opisy do wielu innych gier. **Jakub Kiczemuchowski, ul. Traugutta 2/2 11–500 Giżycko**

Jestem posiadaczem C–16. Poszukuję następujących gier: BRUCE LEE, WEST BANK, SYNTRON i SUPER BOULDER DASH oraz TORPEDO RUN. **Tomasz Praczyk, ul. 22 stycznia 1/20 88–300 Mogilno woj. bydgoskie**

Jak wystartować z grą BLUE MAX, o co chodzi w grach ONE MAN AND HIS DROID i ZORRO II? Poszukuję gier HOBBIT i KINGS OF THE RINGS. Posiadam Atari 65XE i magnetofon. Gry wymienię na inne gry lub zapłacę. **Maciek Puławski, ul. Szewska 19/21 m 7 50–139 Wrocław, tel. 47–24–19**

Mam Atari 800 XL i magnetofon XC 12. Nie wiem, co zrobić w komnacie z komputerem w Universal Hero. Ponadto poszukuję gier Barbarian, Spy vs Spy i The Train. **Małgorzata Rylicka, ul. Słowiańska 13 73-010 Police**

Mam Atari 130 XE. Poszukuję opisów do Ninja Master, Prince of Magic i Super Huey oraz gier Spy vs Spy, The Train. **Paweł Kalinowski, ul. Poniatowskiego 2/1 64-305 Zielona Góra**

Liczę na pomoc czytelników „Bajtka" w zdobyciu gier Barbarian, The Eidolon, Tigers in the Snow, Tour the France na Atari 65 XE. **Zbigniew Cholewa, Dąbrowa St. 3/6 64-200 Wolsztyn**

Poszukuję dokładnego opisu gry Zorro; nie wiem, co zrobić by znaleźć drzwi w podziemiach. Proszę też o podanie sposobu na wpisywanie nieśmiertelności do gier. **Artur Wielgat, ul. Wyspiańskiego 15/20 87-300 Bochnia**

Pomocy! Może ktoś pomoże mi w grze Ace of Aces. Kończę grę i ukazuje się różowa plansza, a ja nie wiem, co dalej. **Zbigniew Kalinowski, ul. Wendy 15 a Gdańsk Osowa**

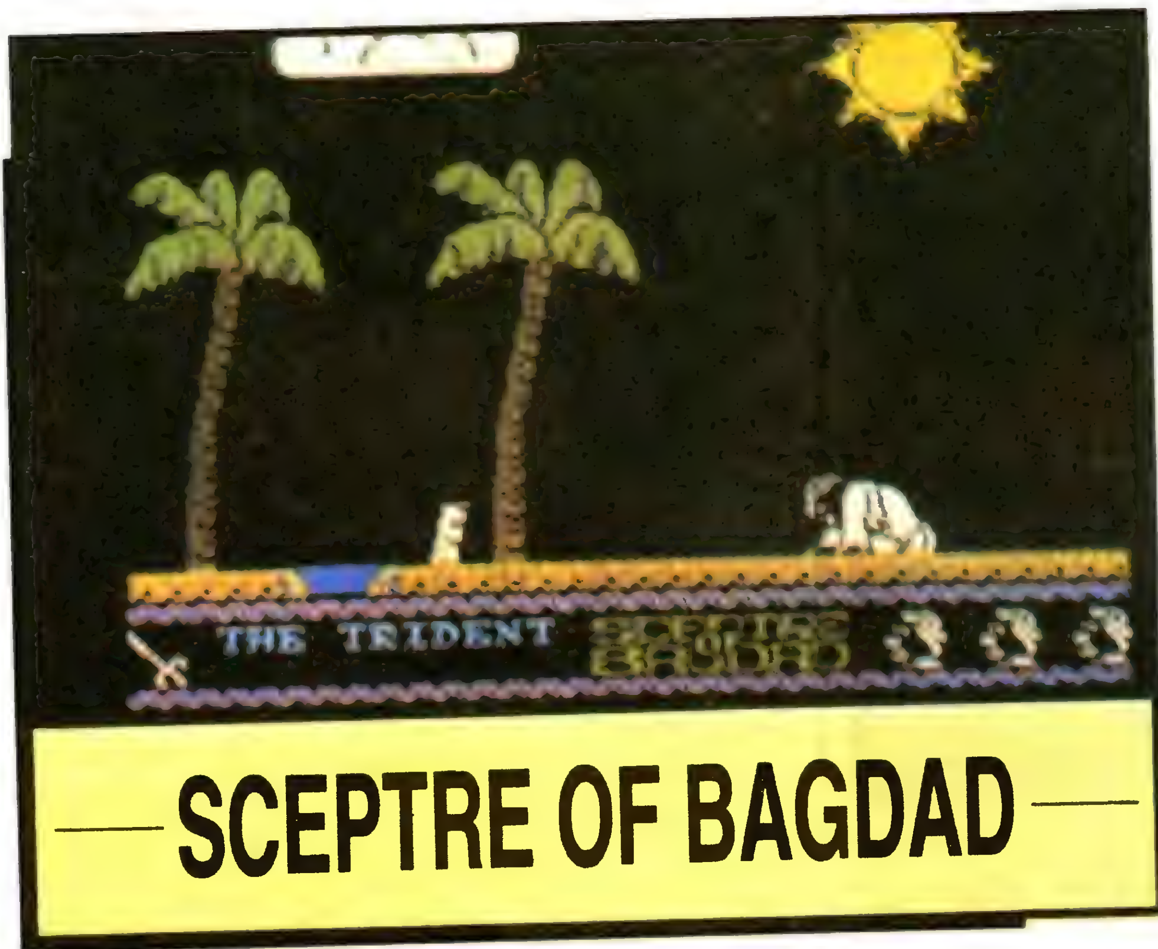
Posiadam Atari 130 XE z magnetofonem. Pilnie poszukuję gier Robocop, Platoon, Barbarian. **Marcin Maziarz, ul. Galileusza 8/8 67-200 Głogów**

Posiadam komputer Commodore 64. Pilnie szukam Aliens, Silent Service. Odstąpię inne gry. **Michał Raczyński, ul. Barana 13/55 42-200 Częstochowa**

Proszę o pomoc w grze The Pawn na Atari XE. Interesuje mnie instrukcja lub odpowiedzi na zadawane w czasie gry pytania. **Cezary Nalewajka, ul. Łódzka 41/5 50-521 Wrocław**

Proszę o przysłanie mi opisów do gier Zorro, ET, Quasimodo, Poker S.A.M. na Atari 65 XE. W zamian wiele opisów **Ewa Słyk, Borek 65 26-931 Zajezierze**

Nigdzie nie znajduję gry Pirates of Barbary Coast w wersji na Atari. W zamian gry Ninja, Spy vs Spy, Henry's House i wiele innych. **Marcin Wilczyński, ul. Mickiewicza 14/31 26-110 Skarżysko-Kam.**



SCEPTRE OF BAGDAD

Ach, te arabskie noce! Duchy, latające dywany, delikatne buciki i inne tureckie rozkosze! Wszystko to możesz znaleźć w zadziwiającej przygodowo-zręcznościowej grze SCEPTRE OF BAGDAD (Berło Bagdadu).

Sułtan Kalif jest zaniepokojony. Pierwszego dnia każdego roku musi pokazać berło miejscowej ludności, aby dowieść, że jest wciąż sułtanem swego kraju. Każdy kto mu w tym przeszkodzi, zostanie powieszony.

Jednakże tego roku Kalif wstał bardzo wcześnie czując się zdecydowanie dziwnie. Szybko wyszedł z łóżka i wskoczył na stół, gdzie stała Czarodziejska Lampa, aby poradzić się ducha. Duch oznajmił, że Kalif padł ofiarą czarów rzuconych przez nikczemnego czarownika. Rezydencja Kalifa została przemieniona w pałac-labirynt pełen rzek z rekinami i palących słońcem pustyń.

Berło znajduje się w najdalszym końcu pałacu. Jeżeli nie zostanie znalezione w określonym czasie, Kalif będzie miał poważne kłopoty — spadnie z tronu. To Ty musisz mu pomóc.

Gracz może poruszać się w prawo i w lewo, skakać przez przeszkody, przechodzić przez drzwi oraz inne wejścia. Przedmioty znalezione po drodze zabierane są przez dotknięcie ich. Jednocześnie nieść można tylko dwa przedmioty. Ich spis będzie pokazany po naciśnięciu spacji.

Poszczególne części pałacu Kalif może przemierzać niosąc odpowiedni przedmiot np. kokos jest potrzebny do przejścia przez pustynię, ale najpierw trzeba zdobyć procę i perłę by dostać się do kokosa.

Kalif wędrując po swym domu spotyka rozmaite potwory i kreatury. Każde dotknięcie ich powoduje utratę jednego z trzech żyć, które ma do dyspozycji gracz. Gdzieś w pałacu ukryta jest butelka z cudownym napojem, po wypiciu której komputer zapamiętuje miejsce, w którym to się stało i w momencie utraty wszystkich żyć można powrócić do tego miejsca wybierając w menu opcję "Old Game".

Każdemu może się przydać trochę arabskich sekretów.

- **FLUTE** — kiedy go masz, możesz wejść po linie w wieży obok drzewa kokosowego
- **WINGS** — daj je statuetce na fontannie, a otworzy się tajemne przejście
- **HAT** — czapka ta spowoduje zniknięcie kobiety w pokoju z lustrem na stole
- **KEYS** — otworzysz nimi ukryte przejście w szafie
- **RING** — otworzy wejście pod wodospadem
- **BOOK** — powoduje zniknięcie głazu broniącego dostępu do groty Ali-Baby
- **WHIP** — gdy jesteś jego posiadaczem wół czuje przed tobą respekt
- **SWORD** — miecz ostry jak brzytwa szybko i skutecznie rozetnie każdą pajęczynę
- **SHOES** — ochronią twe nogi przed rozżarzonym węglem
- **NET** — napełniona siatka na owady jest bardzo przekonywująca dla pająków
- **NURSE** — kiedy jest pełen złota, żaden sklepikarz nie odmówi Tobie wejścia do sklepu

Kiedy już zdobędziesz upragnione berło to czeka Cię tylko krótki spacerek na balkon Twego pałacu.

Firma: Atlantis Software
Komputer: Spectrum, Commodore

Krzychu

Hej, uważaj, piłka już w grze! No, szybciej biegnij do niej! Uff, ledwo zdążyłeś! Tak, już myślałeś, że twórcy gier zapomnieli o twojej ulubionej grze, tenisie, a tu firmy Image Works i Teque sprawiły ci miłą niespodziankę. Po symulacji jazdy od wrotek do czołgu, gry w piłkę nożną i ręczną przyszła kolej na tenis. Ta przeróbka gry z automatów firmy Sega z pewnością rozbawi wszystkich entuzjastów tej starej dyscypliny sportu.

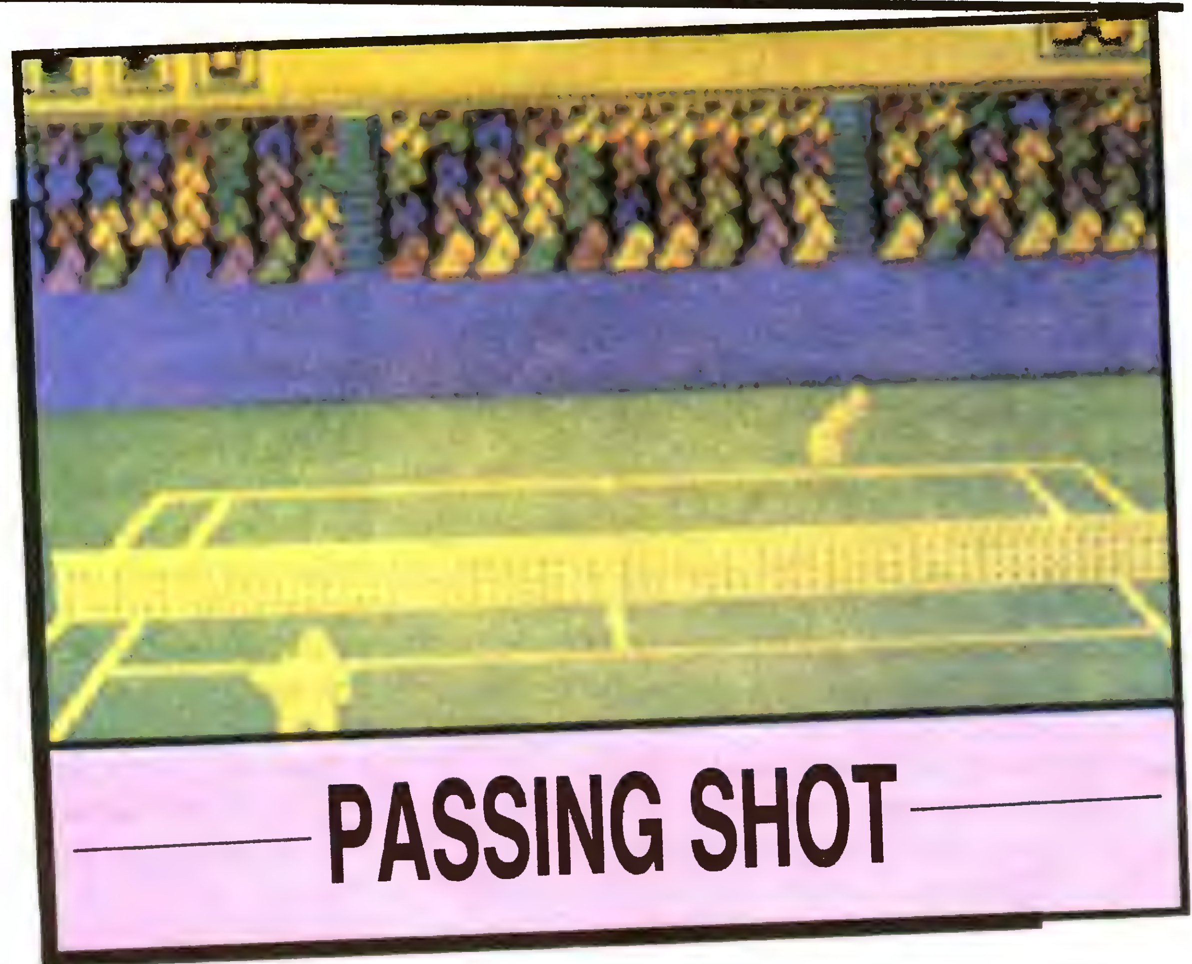
Od razu na początku musisz się zdecydować, w którym z turniejów wielkoszlemowych chcesz uczestniczyć. Masz do wyboru turniej na kortach Rolanda Garosa we Francji (najłatwiejszy), Forest Hills w USA, Flushing Meadows w Australii oraz najtrudniejszy — Wimbledon w Anglii. Passing Shot nie pozwala na zagranie całego meczu: gra zaczyna się w trzecim, decydującym secie. Możesz wybierać między grą singlową i deblową, a przeciwnikiem może być kolega albo komputer. Szkoda, że nie można nazwać zawodnika, którym gra komputer. Można byłoby wtedy wygrać z Lendlem kilka piłek. Ale i tak wszelkie wymagania powinny zostać zaspokojone. W czasie gry są dwie plansze: pierwsza, z poziomu kortu, widoczna jest podczas serwowania piłek, druga pokazuje z góry przebieg samej wymiany piłek, co pozwala łatwo ustawić zawodnika do następnego podania. Ponadto w momencie przyjmowania piłki gracz ma do dyspozycji cztery sposoby jej odbicia: na płasko, ścięcie, górne lub lob. Sprawia to, że gra może wciągnąć, ale mimo wygrania dwóch turniejów nadal nie wiem jak uzyskać każde z odbić! Na samym początku trudne może być nawet poprawne serwowanie: gdy podrzucona piłka zacznie pulsować należy poruszyć się zawodnikiem w górę lub w dół.

Co by nie mówić, jest to jedna z lepszych gier sportowych, chociażby ze względu na doskonałą grafikę: szczegółowe sylwetki graczy, trójwymiarowy kort i dziwny stworek strojący świetne miny po każdym secie. Passing Shot ma też niezłą muzykę, ale już po kilku setach wydaje się, lepiej gdyby jej nie było wcale.

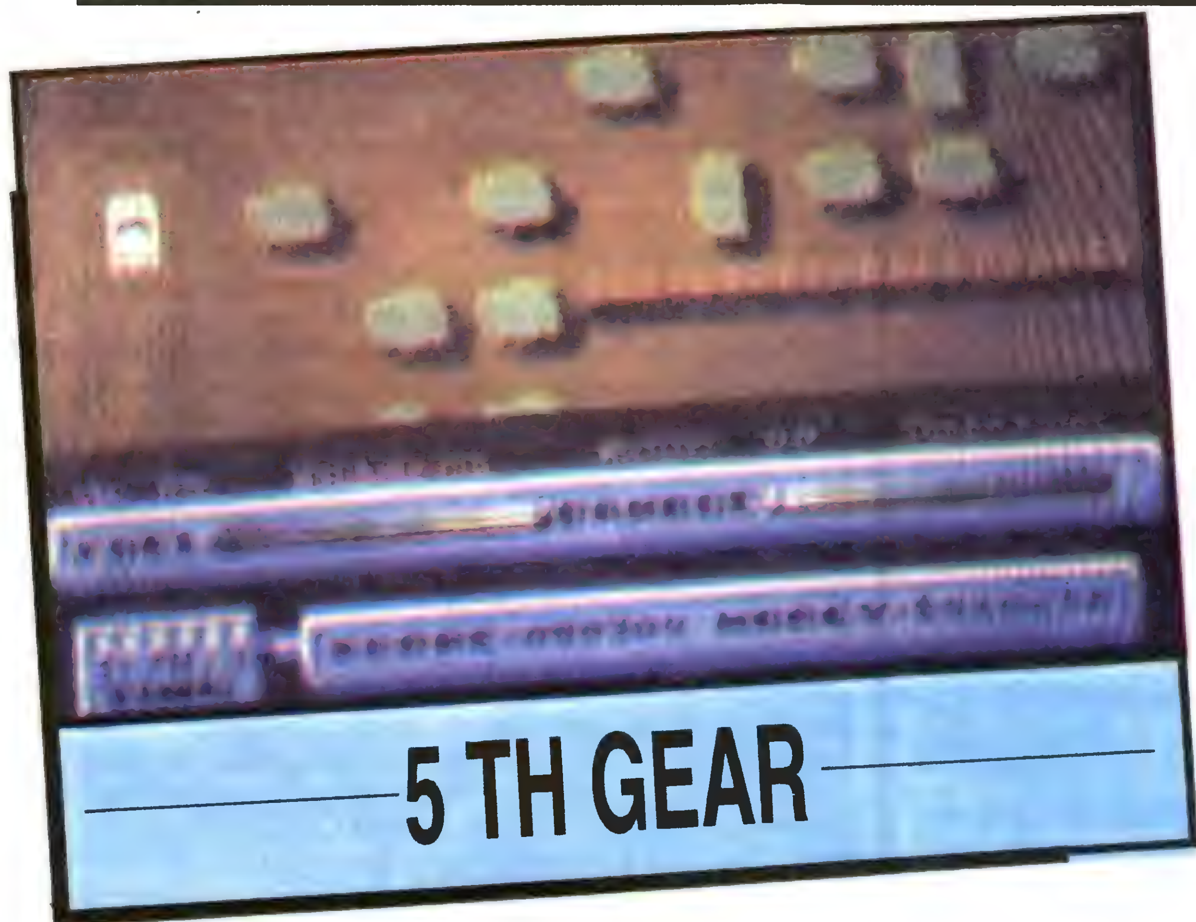
Dla tych, którzy nigdy w tenisa nie grali, ta gra może być nauką podstawowych przepisów, z którymi gra dobrze sobie radzi. Musisz więc wygrać sześć setów, aby wygrać trzeci set, co ci się z pewnością uda po krótkim treningu. Powodzenia.

Firma: Imageworks+Teque
Komputer: Spectrum, Commodore

GIANT



PASSING SHOT



5 TH GEAR

Miał mało czasu. Z każdą sekundą nie wykorzystaną na działanie tracił szansę na przeżycie. Wiedział o tym doskonale i dlatego w kilka sekund zrobił tyle, ile zwykle wykonuje się w minutę.

Wrogów było trzech. Zbliżali się z szybkością stukilkudziesięciu mil na godzinę. Ich samochody, opływowe, o kształtach przypominających wyszlifowane jajka rosły w oczach. Oczyma wyobraźni widział ich głowy w kaskach, zacięte ze złości i nienawiści twarze, ręce na kierownicach i długie, bardzo długie karabiny leżące na siedzeniach. Wiedział, że musiały tam być. Bez nich byliby w stosunku do niego bezbronni, a przecież jechali, by go zabić.

Nie tracił jednak czasu na myślenie o nich. Na to miałby dużo czasu później. Jeśliby przeżył, co nie było wcale takie pewne.

Pierwszą rzeczą jaką musiał zrobić, to uruchomienie samochodu. Potężny Mustang był jego jedyną nadzieją. Nisko pochylona sylwetka i podwójne koła z tyłu dawały doskonałą przyczepność. Rozbudowany według własnego pomysłu silnik, zwany Garrym, pozwalał na osiągnięcie największych na świecie szybkości. Tylko samoloty i odrzutowe helikoptery mogły mu zagrozić. Ale i dla nich miał dużo niespodzianek. Ukryte działko przeciwlotnicze kierowane oczywiście komputerem, specjalna substancja rozpylana w powietrzu i zakłócająca pracę silników samolotowych — to walory, o których nie można zapominać przy rozpatrywaniu wzajemnych szans.

Gdy samochód stanął gotowy przed bunkrem, będącym jego garażem, podbiegł jeszcze do skrzynki w ścianie i wcisnął dwa guziki. Bunkier zanurzył się w ziemi. Uśmiechnął się, gdy wyobraził sobie miny jakie będą mieli, kiedy się tu zjawią.

Z całą siłą wcisnął pedał akceleratora. Przyspieszenie wgniotło go w supermiękki fotel, krępując ruchy i zwalniając pracę mózgu. Nie przejmował się tym. Musiał odjechać na odpowiednią odległość by móc walczyć. Jego samochód przypominał myśliwiec który bez przewagi wysokości nad przeciwnikiem miał małe szanse na zwycięstwo.

Pierwsi zaatakowali oni. Potrójna salwa pocisków ziemia — ziemia z oszałamiającą prędkością zaczęła zbliżać się do jego samochodu. Natychmiast wystrzelił sześć antyrakiet — po dwie na każdy pocisk. Trzy potężne wybuchy wstrząsnęły po chwili powietrzem. Trzy niewielkie słońca zapaliły się na moment na niebie, by po chwili zgasnąć i zamienić się w dymiące kule.

Postanowił nie czekać dłużej. Wcisnął guzik z napisem „laser-gun" i zmienił kierunek jazdy o 180 stopni. Widział teraz ich w przedniej szybie. Po wciśnięciu dwóch kolejnych przycisków, szyba zamieniła się w wielki panoramiczny celownik. W samym jego środku znajdował się jeden z pojazdów. Komputer automatycznie wystrzelił w jego kierunku wiązkę śmiertelnego światła. Po trafieniu samochód dosłownie zniknął, nie został po nim żaden ślad.

Drugi z przeciwników zginął od własnej broni. Rakieta wystrzelona przez niego miała najprawdopodobniej wadliwie ustawiony układ kierowania i zaraz po starcie wybuchła. Siła eksplozji zniszczyła wszystko w promieniu dwudziestu metrów, łącznie z jej właścicielem.

Ostatni akt dramatu rozegrał się nad jeziorem. Nieskuteczne wymiany strzałów zaczęły już nużyć obydwu pozostałych przy życiu kierowców. I właśnie w momencie, gdy jeden z nich zamierzał rozstrzygnąć wszystko rozpylając zasłonę dymną, silnik trzeciego z napastników nie wytrzymał. Spod maski zaczął wydzielać się czarny, gęsty dym, potem jaskrawy płomień palącej się ropy, większy i większy, aż wreszcie... Wybuch paliwa dokończył dzieła zniszczenia.

Trzech ich było i trzej zginęli...

Często zdarza się jednak, że za kierownicą samochodu nie zasiądzie tak wytrawny kierowca, jak to miało miejsce tym razem. Obserwujemy wtedy zjawisko bardzo zabawne — jeden samochód na sekundę kończy swoje życie. Prosimy więc, aby do gry 5th Gear nie siadali ślepi, kalecy i daltoniści. Jest to program tylko dla zawodowców i pytanie „być albo nie być", nie ma tu prawa bytu.

Komputer: Spectrum, Commodore

LUKE

Mała rzecz, a cieszy — czyli o własnościach portu F8

Gdy pewnego dnia z nudów przyglądałem się zawartości pliku J14CPM3.EMS, rzuciły mi się w oczy częste odwołania do portu o adresie F8. Okazało się, że intuicja mnie nie zawiodła — daje on programiście sporo nowych możliwości.

Po pierwsze: instrukcja **port[$F8]:=1** (lub **OUT &HF8,1** w BASIO-u) daje ten sam efekt, co naciśnięcie jednocześnie klawiszy SHIFT, EXTRA i EXIT, czyli resetuje komputer.

Po drugie: **port[$F8]:=8** gasi ekran, a **port[$F8]:=7** ponownie przywraca wyświetlanie. Można to wykorzystać przy odczytywaniu ekranu z dyskietki wyłączając go przed operacją i włączając dopiero po jej zakończeniu. Zamiast stopniowego pojawiania się obrazka, ujrzymy go od razu w całości.

Po trzecie (chyba najciekawsza możliwość): **port[$F8]:=11** uruchamia głośnik, a **port[$F8]:=12** wyłącza go. Na tej samej zasadzie, co w ZX Spectrum, można tworzyć dźwięki znacznie ciekawsze niż pisk uzyskiwany przez **write(H7)** (lub **PRINT CHR$(7)**), chociaż, ze względu na słabą jakość głośniczka, gorsze od tych z komputerka Sinclaira.

W celach eksperymentalnych przedstawiam dwie procedury pascalowe. Pierwsza z nich, **SOUND(dl,wys,gl:integer),** wydaje dźwięk o długości **dl,** wysokości **wys** i głośności **gl.** Druga procedura, **Q(n:integer),** dla n=1,2 daje dwa ciekawe efekty dźwiękowe.

Mam nadzieję, że dzięki tym informacjom ktoś inny wydusi z **Amstrada PCW** ciekawsze dźwięki niż mnie się to udało.

Andrzej Wawrzyńczyk

```
procedure Sound (dl,wys,gl : integer);
begin
  inline(237/$5B/dl/$2A/wys/$2B/$7C/$B5/32/251/62/11/211/$F8/237/
        75/gl/11/$78/$B1/32/251/62/12/211/$F8/27/$7A/$B3/32/226);
end; (* of Sound *)

procedure Q (N : byte);
begin
  case n
  of  1 ; inline(62/12/6/240/14/254/37/32/6/238/7/211/$F8/38/238/
               45/32/244/238/7/211/$F8/46/254/16/236/62/12/211/$F8);
      2 ; inline(17/98/100/38/10/62/11/254/11/40/3/61/24/1/60/211/
                $F8/67/16/254/37/32/240/28/21/32/232/62/12/211/$F8);
  end;
end; (* of Q *)

var
   ch : char;

begin
  Q(1);
  repeat until KeyPressed; read (kbd,ch);
  Q(2);
  repeat until KeyPressed; read (kbd,ch);
  Sound (50,400,100);
end.
```


TIMEX bez tajemnic (2)

W poprzedniej części opisałem różnice pomiędzy Spectrumną i Timexem związane z grafiką. W uzupełnieniu warto podkreślić, że obsługę wszystkich dodatkowych trybów grafiki Timexa twórcy mikrokomputera całkowicie pozostawili użytkownikowi. Dotyczy to takich instrukcji jak np. PRINT, LIST, PLOT, DRAW, CIRCLE. Działają one poprawnie tylko dla OUT 255,0. Na pierwszy rzut oka jest to wada: prawie nikt nie korzysta z tych dodatkowych trybów, bo musiałby napisać własne, wymienione wyżej, procedury w asemblerze. Mimo to jest to też zaleta: pozostawiono prawie niezmieniony ROM ZX Spectrum (jedyną różnicę podałem w cz. 1). Dzięki temu na Timexie poprawnie działa ponad 95% programów na Spectrumnę. Ale czemu pozostałe nie działają? Otóż niektóre programy (przede wszystkim gry) lubią „śmiecić" różnymi niepotrzebnymi instrukcjami (może nią być np. OUT 255, nr itp.). Jeżeli w ten sposób, zmieni się na moment tryb graficzny to jeszcze nic, gorzej gdy nr=64 co wyłączy przerwania i program się „zawiesi". Niestety to nie jest jedyny powód dziwnego działania programów. Następną przyczyną może być obsługa klawiatury.

Klawiatura w Timexie

Listy Czytelników dowodzą, że w tej kwestii panuje niezły bałagan. Po pierwsze klawiatura minimalnie różni się z wierzchu, ale połączenia wewnętrzne odpowiadają połączeniom wewnątrz ZX-a. Po drugie porty klawiatury naprawdę są takie same jak w ZX Spectrum. Przypomnę, że ich numery opisuje wzór IN (256∗(255-2^n)+254) gdzie n=0, 1, .., 7. Każdy z ośmiu portów opisuje stan pięciu klawiszy. Każdy z nich, z kolei, jest opisany pojedyńczym bitem. Są to bity 0–4. Bit równy zero oznacza, że klawisz jest wciśnięty. Bit nr 6 opisuje stan gniazda EAR. Niespodzianką jest fakt, że niewykorzystane bity 5 i 7 oraz bit 6 mają odwrotną wartość niż w ZX Spectrum. Jest to główną przyczyną złego działania programów, które samodzielnie, z pominięciem procedur z ROM czytają stan klawiatury. Źle działają tylko te, które zamiast sprawdzać pojedyńcze bity, porównują łączną zawartość portu klawiatury z wzorcem. Poniższa tabela powinna wyjaśnić te różnice.

numer portu klawiat.	wartości zwracane przez ZX Spectrum				
	175/239	183/247	187/251	189/253	190/254
IN 32766 #7FFE	B	N	M	S.SHIFT	BREAK SPACE
IN 49150 #BFFE	H	J	K	L	ENTER
IN 57342 #DFFE	Y	U	I	O	P
IN 61438 #EFFE	6	7	8	9	0
IN 63486 #F7FE	5	4	3	2	1
IN 64510 #FBFE	T	R	E	W	Q
IN 65022 #FDFE	G	F	D	S	A
IN 65278 #FEFE	V	C	X	Z	CAPS SHIFT
gniazdo EAR nakt/akt	79 / 15	87 / 23	91 / 27	93 / 29	94 / 30
	wartości zwracane przez Timex				

Przez tę zmianę gry tak znane jak np. URIDIUM firmy HEWSON samodzielnie czytające klawiaturę i joystick nie chcą poprawnie działać. W przypadku tej gry wystarczyło po rozpoznaniu, że została ona uruchomiona na Timexie zmienić wartość jednego wzorca. Przykładowy program wyglądałby następująco:

```
0 REM URIDIUM na Timexa
5 LOAD "" CODE
10 IF (IN 254 < 128) THEN POKE
  43785,17
15 RANDOMIZE USR 64848
```

Warunek w linii 10 jest spełniony tylko na Timexie (dla ZX Spectrum komórka 43785 powinna zawierać jedynkę). Podobne problemy występowały w grach FIRE LORD, TOP GUN itd. Niektóre gry, pomimo błędnej obsługi klawiatury, dają się uruchomić na joysticku. Ale z joystickiem też należy być ostrożnym. Poniższe uwagi dotyczą na szczęście tylko joysticków pracujących w trybie KEMPSTON. Otóż Timexy są wyposażone fabrycznie w interfejs KEMPSTON i jest to cenne rozszerzenie ich możliwości. Pozwala on oszczędzić podczas gier niezbyt wytrzymałą klawiaturę. Jak zapewne wszyscy wiedzą, jest on dekodowany rozkazem IN 31. Niestety w tym przypadku twórcy Timexa niepotrzebnie uprościli sobie życie i wzorem Sinclaira zastosowali tzw. dekodowanie niepełne. Ta nazwa oznacza tyle, że podczas testu joysticka wystarczy podać dowolny adres, którego bit A4 jest równy zeru. Może to zatem być IN 31, IN 47 nawet IN 0. Takie dekodowanie jest dużo prostsze w realizacji niż dekodowanie pełne, ale marnuje się w ten sposób wiele portów we/wy. W szczególności użytkownik Timexa nie może podłączyć do swego mikrokomputera żadnego interfejsu, który zwraca do komputera jakieś dane po rozpoznaniu rozkazu IN xxx0xxx (binarnie, gdzie x oznacza 0 lub 1), ponieważ będzie on źle działał. Posiadacze ZX Spectrum są w lepszej sytuacji bo mogą wyjąć interfejs KEMPSTON i już jest po kłopocie.

W następnej części zajmę się pamięcią RAM Timexa i opiszę jak w zwykłym Timexie zamontować dużo większą pamięć RAM np. 80 KB zamiast 48 KB albo przeciwnie, odłączyć część już nie istniejącego RAM-u. Zachęcam do poeksperymentowania z już poznanymi sztuczkami i do napisania na ich podstawie własnych programów na Timexa, bo następne sztuczki będą dla bardzo zaawansowanych Czytelników.

Marek Sawicki



Kto próbował już sił w programowaniu w języku wewnętrznym przyzna, że dużo racji jest w stwierdzeniu, iż umiejętność programowania tak się ma do znajomości listy rozkazów, jak umiejętność mówienia w danym języku do znajomości słownika.

Wiąże się to z faktem, że każdy rozkaz z bogatej listy rozkazów procesora jest prymitywną operacją, której wykonanie niczego nie rozwiązuje.

Na rozwiązanie określonego problemu mamy szansę dopiero wtedy, gdy użyjemy dużej liczby rozkazów. Pojęcie „duża" jest tu umowne i zależy od stopnia złożoności rozwiązywanego problemu.

Rola programisty jest tu absolutna i nikt, a komputer na pewno, wyręczyć go nie może. Od komputera zależą natomiast generalnie dwie sprawy: zawrotna szybkość wykonywania poszczególnych operacji oraz ich bezbłędna realizacja.

To sprawia, że nawet bardzo złożone problemy, których rozwiązanie wymaga użycia bardzo dużej liczby rozkazów realizowane są w ułamkach sekund.

Z punktu widzenia czasu realizacji programu nie ma znaczenia, czy do rozwiązania zagadnienia użyjemy 1000 czy 2000 rozkazów, gdyż oba programy wykonane będą w porównywalnie krótkim czasie. Nie może to być natomiast obojętne z punktu widzenia wykorzystania pamięci oraz efektywności algorytmu.

Oprócz wysokiej specyfiki i efektywności działania programów, język wewnętrzny umożliwia maksymalne wykorzystanie pamięci i możliwości procesora, dzięki czemu z podrzędnego komputera z jego językiem wysokiego poziomu otrzymujemy maszynę o lepszej klasie i szerszym zastosowaniu.

Warto więc poświęcić trochę czasu, by poznać „tajemnice" języka maszynowego i nauczyć posługiwania się nim choć w stopniu podstawowym. Nie jest to jednak łatwe, gdyż potrzebna jest umiejętność ułożenia algorytmu działania. Powinien on umożliwiać rozwiązanie zagadnienia przy użyciu dostępnych rozkazów procesora, często nie mających wiele wspólnego z rozwiązywanym problemem. W sumie sprowadza się to do ciągłego transponowania sposobu myślenia człowieka na sposób myślenia maszyny. Niezbędna jest tu pełna znajomość systemu operacyjnego i jego procedur, mapy pamięci itp. oraz umiejętność zakodowania algorytmu, czyli napisania programu w języku maszynowym.

Aby umieć programować, trzeba koniecznie przyswoić sobie istotę algorytmu i sposób jego tworzenia. Natomiast znajomość systemu, procedur itp. może być okrojona, lecz nie zarzucona.

Przechodząc do konkretów posłużmy się przykładem. Zadaniem do rozwiązania niech będzie napisanie na ekranie cyfry „1".

W języku wysokiego poziomu, np. Sinclair Basic „powiemy" komputerowi „Pisz na ekranie znak 1". Sprecyzujemy miejsce napisania znaku, tj. przecięcie wiersza w i kolumny k ekranu (rozdzielczość 32×24 znaki). Instrukcja będzie sformułowana dokładnie w „języku człowieka":

PRINT AT w,k;1

Natomiast w języku wewnętrznym ta czynność nie będzie miała nic wspólnego ze sposobem myślenia człowieka, albowiem pojęcie „pisz w określonym miejscu" rozumiane jest „załaduj" (wypełnij) do komórek pamięci odpowiedzialnych za dany wycinek ekranu określone bajty, z których na ekranie powstanie znak „1".

Z tą czynnością wiąże się oczywiście konieczność odpowiedzenia na wiele pytań: jakie bajty? skąd je wziąć? dokąd je przesłać? które komórki pamięci (adresy) odpowiadają za pozycję na przecięciu wiersza w i kolumny k? i nie tylko.

Z przytoczonego przykładu widać zasadniczą różnicę między programowaniem w języku wysokiego poziomu a programowaniem w języku wewnętrznym. Programowanie w języku wysokiego poziomu to mozolna praca realizowania algorytmów z funkcjonalnie oczywistych instrukcji, zaś w języku wewnętrznym programowanie to intelektualna twórcza działalność programisty, poparta opanowaną do perfekcji znajomością sprzętu oraz arytmetyki logicznej i binarnej.

Wykonawca zlecanych mu operacji, procesor, jest urządzeniem elektronicznym. „Rozumie" więc jedynie sygnały elektryczne, które dla maksymalnego uproszczenia ograniczono do dwóch stanów, tzn. „jest napięcie" i „nie ma napięcia". Pierwszy stan oznaczono cyfrą 1, zaś drugi cyfrą 0. Ciąg takich sygnałów można więc nie tylko mierzyć i obserwować na oscyloskopie, ale i zapisać.

Znak 0 lub 1 to najmniejsza porcja informacji, zwana bitem. Osiem bitów tworzy najmniejsze powszechnie spotykane słowo informacji — bajt. Z ośmiu bitów zgodnie z teorią kombinatoryki matematycznej można zestawić 2^8 kombinacji, co w dwójkowym systemie liczenia daje 256 liczb, od 0 do 255.

Procesor Z80 został tak skonstruowany, że rozróżnia i realizuje ok. 700 różnych czynności zwanych rozkazami. Każdy rozkaz posiada swój kod, który jest liczbą lub ciągiem liczb, po którym procesor rozpoznaje, jaką czynność ma wykonać. Ponieważ liczby 256 przekroczyć nie można, więc by zakodować wszystkie 700 rozkazów, niektóre liczby kodujące powtarzają się.

Aby uniknąć niejednoznaczności, wydzielono 4 liczby (203, 221, 237 i 253), które samotnie nie kodują żadnego rozkazu. Nazwano je prefiksami (w jęz. angielskim — przedrostek) i poprzedzając kody zestawu podstawowego tworzą one kody nowych rozkazów. Dla przykładu liczba 70 wchodzi w skład kodów aż 7 rozkazów.

Tak rozumiane kody rozkazów wraz z ich argumentami to właśnie elementy języka wewnętrznego procesora, czyli języka maszynowego. Ciąg liczb umieszczonych w pamięci i stanowiących zestaw logicznie ułożonych kodów kolejnych rozkazów realizujących określone zadanie nazywa się procedurami maszynowymi lub programem w języku maszynowym (wewnętrznym).

Pisanie programów przy użyciu kodów oraz dalszych liczb będących argumentami rozkazów jest niezmiernie uciążliwe. Celem ułatwienia programowania kody rozkazów zastąpiono nazwami odpowiednich czynności (tzw. mnemonikami), zapisanymi w skrócie. Podobnie argumenty tych rozkazów zapisano nie tylko przy użyciu liczb, lecz i określonych symboli, co jest konsekwencją tego ułatwienia.

Tak spreparowany język wewnętrzny to właśnie język asemblera, czyli krótko asembler.

Program asemblerowy realizujący nasz problem będzie miał postać na przykład taką:

```
10  ADRES   DEFW 8874        ;procedura znajdująca adres danego
                             ;bajtu na ekranie
20  ADRGEN  DEFW 15752       ;adres znaku „1" w generatorze znaków
30  LICZ    DEFB 8           ;licznik ośmiu bajtów
40  IKS     DEFB 120         ;wsp. X środka ekranu (pixele)
50  IGREK   DEFB 87          ;wsp. Y środka ekranu (pixele)
60          ORG  60000       ;adres w pamięci do umieszczenia
                             ;kodu wynikowego
70          LD   C,IKS       ;do rejestru C załaduj wartość X
80          LD   B,IGREK     ;do rejestru B załaduj wartość Y
90          CALL ADRES       ;ustal adres tego bajtu na ekranie
100         LD   B,LICZ      ;do B załaduj wartość licznika (8)
110         LD   DE,ADRGEN   ;do DE załaduj adres gen. znaku „1"
120 PETLA   LD   A,(DE)      ;do akumulatora załaduj bajt z
                             ;pamięci o adresie zapisanym w DE
130         LD   (HL),A      ;przeładuj zawartość A do pamięci
                             ;ekranu pod adres wskazany w HL
140         INC  H           ;przejdź do nast. bajtu ekranu (o 256)
150         INC  E           ;przejdź do nast. bajtu generatora zn.
160         DJNZ PETLA       ;zmniejsz B o 1 i powtórz pętle,
                             ;gdy B<>0
170         RET              ;powróć do Basica
```

Program pisany jest w określonej postaci, tzn. każda linia ma swój numer porządkowy zapisany w pierwszym od lewej strony polu. Następnie po spacji rozpoczyna się pole etykiet, składające się z sześciu znaków, zakończone także spacją. Etykieta to symbol złożony ze znaków ASCII, przy czym pierwszy znak musi być literą. Taki format przyjmuje GENS, o którym za chwilę.

Z punktu widzenia przejrzystości programu korzystne jest, by etykieta określała nazwę lub skrót nazwy, której dotyczy. W naszym programie mamy więc etykiety, które już wstępnie informują, czego dotyczą. Na przykład ADRES i ADRGEN informują, że pod nimi kryje się liczba będąca adresem w pamięci, w drugim przypadku adresem generatora. LICZ to licznik, IKS i IGREK — współrzędne, zaś PETLA to podprogram realizujący pewną pętlę.

Szersze wyjaśnienie znaczenia poszczególnych linii, jeśli potrzebne, można umieścić w polu zwanym polem komentarzy, które musi być poprzedzone znakiem średnika ";".

Pomiędzy polem etykiet a polem komentarzy znajdują się dwa pola rozdzielone spacją, a mianowicie pole mnemoników i pole argumentów. Pole mnemoników to czteroznakowa przestrzeń, gdzie wpisywana jest symboliczna nazwa rozkazu, zaś pole argumentów, jeśli istnieje, zawiera argument liczbowy rozkazu lub symbol argumentu. Wielkość tego pola nie jest sprecyzowana i zależy od argumentu. Jeśli np. przyjmiemy etykiety dłuższe niż sześć znaków lub zapiszemy liczby w postaci dwójkowej (poprzedzone znakiem "%") lub szesnastkowej ("#"), pole wydłuży się.

Przestrzeganie tych zasad ułatwia pisanie programów, lecz by dany program mógł być zrealizowany przez komputer, trzeba go zamienić na kod maszynowy i dopiero w takiej postaci uruchomić. Kodowanie programu można wykonać ręcznie, w oparciu o listę rozkazów, jednak przy programach dłuższych byłoby to niezmiernie uciążliwe.

Aby maksymalnie ten proces uprościć, opracowane zostały specjalne programy — narzędzia. Jest ich kilka rodzajów i wersji różnych firm. W Polsce powszechnie dostępnymi są GENS i MONS firmy Hisoft. GENS jest to asembler zintegrowany z edytorem, przy jego pomocy można dokonywać asemblacji, czyli tłumaczenia programu z asemblera (tekst źródłowy) na kod maszynowy (kod wynikowy) oraz edycji, tzn. wpisywania i poprawiania tekstu programu w asemblerze.

Drugi program — MONS — służy do uruchamiania programów, ich testowania i przeglądania oraz do zamiany kodu maszynowego na tekst w asemblerze (deasemblacja).

Naukę asemblera rozpoczyna się zwykle od mniej lub bardziej szczegółowego omówienia budowy i zastosowania wszystkich rozkazów procesora. My pominiemy ten etap a doskonalenie umiejętności programowania spróbujemy realizować inaczej, a mianowicie przez analizę wielu gotowych procedur takich, by z jednej strony pokazać zastosowanie wielu rozkazów, z drugiej zaś zwrócić uwagę na spotykane układy rozkazów i sposoby ich wykorzystania.

Za podstawę przyjmiemy wybrane procedury z ROM-u Spectrum starając się, aby równolegle poznawać ich funkcje i zastosowanie w systemie. Ponadto wydaje się, że właśnie Spectrum stanowi doskonałą bazę, albowiem jest na tyle dostępny, że każdy ma szansę jego poznania i dzięki temu wkroczenia w świat na początek małych, później dużych komputerów.

Prezentowane opracowania kierowane są głównie do amatorów, którzy nie mając możliwości studiowania lub uczestniczenia w kursach specjalistycznych poznali Basic na tyle, że przestało to im już wystarczać i chcieliby sięgnąć do asemblera. W tym miejscu zaczyna się charakterystyczny problem polegający na tym, że studiuje się listę rozkazów, ale przez długi czas nie potrafi się żadnego z nich zastosować. Dlatego też ten artykuł wprowadzający i

następne mają na celu pomóc ten problem szerzej niż amatorsko rozwiązać.

Zanim rozpoczniemy analizę poważniejszych procedur, spróbujmy tytułem próby przeanalizować zapisany wyżej program w sposób, jaki chcielibyśmy dalej stosować.

1. **Cel** — napisać na środku ekranu cyfrę "1".
2. **Istota algorytmu** — adres bajtu 11 wiersza 16 kolumny (środek ekranu) ustalić wykorzystując procedurę z ROM-u (8874), adresy pozostałych siedmiu bajtów znaku różnią się o 256.
3. **Wzór znaku "1"** — wykorzystać wzór systemowy zapisany w ośmiu bajtach generatora znaków poczynając od adresu 15752 (adres pierwszego bajtu 96 znaków ASCII wynosi 15616 — spacja, ósmego bajtu ostatniego znaku — 16383).
4. **Opis działania programu** — program można podzielić na trzy zasadnicze części: zdefiniowanie adresów i danych (10—60), ustalenie danych początkowych w rejestrach (70—110) i zapętlone przeładowanie ośmiu bajtów znaku "1" z obszaru generatora znaków do pamięci ekranu. W wyniku tego na ekranie pojawi się cyfra 1.

W pierwszym etapie kolejno zdefiniowano:

— przy pomocy dyrektywy DEFW (define words) definiującej liczby dwubajtowe określono adres procedury w ROM, która na podstawie współrzędnych x i y (jak dla PLOT) oblicza adres odpowiadającego bajtu na ekranie. Dyrektywie nadano etykietę ADRES.

— w linii 20 adres pierwszego bajtu jedynki w generatorze znaków przyporządkowano dyrektywie ADRGEN.

— pod etykietą LICZ liczbę 8 — liczbę obiegów pętli.

— pod IKS i IGREK — współrzędne środka ekranu. Tu uwaga: współrzędna y przy tak przyjętym algorytmie musi mieć wartość będącą wielokrotnością ósemki minus 1, co wynika z organizacji ekranu w Spectrum. W przeciwnym razie nasz znak może zostać wypisany w dwóch częściach.

— przy użyciu ORG — adres, gdzie umieścić kod wynikowy, przeznaczony do uruchomienia. Jest to wytyczna dla programu GENS.

Procedura, której adres określono w 10 wymaga umieszczenia w rejestrach C i B odpowiednio odciętej i rzędnej punktu. Rozkaz 90 wykonuje skok do procedury, która szukany adres umieszcza w rejestrze HL. Po powrocie linie 100 i 110 umieszczają w B liczbę 8 a w DE adres pierwszego bajtu znaku w generatorze. Są to wystarczające dane, aby rozpocząć przepisywanie bajtów z generatora znaków do pamięci ekranu przy równoczesnym liczeniu, by przepisać tylko 8 bajtów znaku "1".

Główna pętla rozpoczyna się linią 120, która przesyła do akumulatora zawartość komórki pamięci, której adres znajduje się w DE. Zapis (DE) oznacza, że daną należy pobrać nie z rejestru, lecz z pamięci spod adresu zapisanego w rejestrze DE. Tak więc do A zostaje załadowany pierwszy bajt znaku "1" z generatora.

Linia 130 ten bajt ładuje do komórki pamięci, której adres znajduje się w HL, tzn. do komórki określającej górny bajt interesującego nas kwadracika na ekranie. Przeładowanie musi odbyć się w ten sposób, gdyż brak jest rozkazu typu LD (HL), (DE).

Kolejne linie dokonują zwiększenia o 1 rejestru H (starszy bajt zawartości rejestru HL), stąd zawartość HL zwiększa się o 256 (przejście o linię niżej na ekranie) oraz zwiększenia o 1 zawartości rejestru E, stąd zawartość DE zwiększa się również o 1 (następny bajt generatora).

Na uwagę zasługuje rozkaz DJNZ w linii 160, który szczególnie nadaje się do sterowania pętlą, w której liczba obiegów nie przekracza 256. Jest to skok warunkowy do etykiety, przy czym warunkiem skoku jest „nie zero" (NZ) w rejestrze B, który przed każdym skokiem automatycznie zmniejszany jest o 1.

Program zamieniony na kod wynikowy (dyrektywa A w GENS-ie) ma postać:

14,120,6,87,205,170,34,6,8,17,136,61,26,119,36,28,16,250,201

Te 19 liczb umieszczone jest kolejno w pamięci od adresu 60000 do 60018 i stanowi program gotowy do bezpośredniej realizacji przez procesor. Kontrolę nad prawidłowością merytoryczną i formalną przejmuje tylko programujący.

Program uruchamia się przez RANDOMIZE USR 60000.

Piotr Sumara

Literatura:
1. J. Kaczmarczuk — Mikroprocesor Z80
2. Konkret, zeszyt 3—11
3. K. Sacha, A. Rydzewski — Mikroprocesor w pytaniach i odpowiedziach
4. K. Kuryłowicz, D. Madej, K. Marasek — Przewodnik po ZX Spectrum

SYNTHSET - cz II

Przedstawiam dwie procedury, służące do odtwarzania dźwięku, zapisanego za pomocą programów tworzących samplingi. Jeden z takich programów znajduje się w części pierwszej artykułu (Bajtek 5-6/90).

Na listingu nr 1 znajduje się procedura odtwarzająca dane zapisane i stworzone za pomocą SYNTHESET'a.

Listing programu jest w kodzie maszynowym. Należy wpisać go przy pomocy asemblera (najlepiej GENS'a).

Procedura ta może być w dowolnym miejscu pamięci, lecz należy pamiętać, że umieszczenie poniżej adresu **32768** spowolni jej działanie. Zmianę adresu startowego procedury dokonujemy przez wstawienie odpowiedniej wartości do linii **80**.

Głośność korygujemy poprzez zmianę wartości akumulatora w linii **150**. Należy przy tym pamiętać, że może to być wartość z zakresu 1—15.

Oczywiście najważniejsze jest to, aby program wiedział skąd pobierać dane i jaka ma być ich ilość. Te wartości możemy mu podać w liniach **300** i **310**. W linii **300** podajemy do rejestru HL początek danych, a w linii **310** do rejestru DE ilość.

Bardzo istotną jest wartość prędkości z jaką odtwarzamy sampling. Wartość tą możemy zmieniać przez ustawienie opóźnień w naszej procedurze. Czynimy to przez zmianę wartości rejestru C w liniach **330** oraz **580**. Opóźnienia te winny spełniać wzór:

DELAY1 = **DELAY2** + **2** albo **DELAY2** = **DELAY1** − **2.**

W rzeczywistości oznacza to, że opóźnienie DELAY2 powinno być o dwa mniejsze od opóźnienia DELAY1, lecz nie powinno być zerem.

Przyjmując powyższe założenia minimalną wartością DELAY1 jest trzy, a tym samym minimalną wartością DELAY2 jest jeden.

Maksymalne wartości to teoretycznie 255 i 253, lecz praktycznie liczby oscylujące w granicach 25 do 30 (z uwagi na wyrazistość odtwarzania).

Teraz krótka analiza naszej procedury.

Linie od **100** do **280** inicjują wstępnie rejestry AY-greka, linie od **300** do **620** zawierają główną pętlę odtwarzającą, przy czym linia **360** testuje bity kolejnych bajtów zawierających dane i odpowiednio wykonywany jest skok do części NOPE jeśli bit był zerem. W przypadku gdy bit był zapalony (ostrożnie z ogniem) zostaje wykonana sekwencja rozkazów uaktywniających rejestry AY-greka.

Linia **620** włącza przerwania, i powrót z naszej procedury następuje w linii **630**.

Procedura po kompilacji ma długość około 110 bajtów. Ważne jest aby dane były w pamięci pod adresem zgodnym z tym, który wpisaliśmy do naszej procedury, bo w przeciwnym przypadku zamiast muzyki, mowy albo śpiewu (danych) możemy usłyszeć tylko nieprzyjemny szum.

Wielu z Was, Drodzy Czytelnicy, na pewno by chciało poeksperymentować z profesjonalnie zapisanym samplingiem — takim jak np. w grach **ROBOCOP, HELFIRE ATTACK, CHASE H.Q.**

Na listingu nr 2 przedstawiam Wam procedurę do odtwarzania samplingów z większości gier. Procedurę bliźniaczą do tej użyłem w swoich programach: **RAVEL DEMO, MAGNIFICENT LAUGH** oraz w **SABRINA DEMO 48/AY**.

Za pomocą tej magicznej procedury możecie przeszukiwać miliardy bajtów swoich programów, w których to może zapodziały się zapomniane sample autorów, albo takie, które uruchamiają się tylko na Spectrum 128.

Z tą procedurą postępujemy podobnie jak z poprzednią, lecz zmian dokonujemy tylko w trzech liniach. W linii **60** zmieniamy długość, w linii **70** początek danych a w linii **420** możemy ustawić opóźnienie.

Dźwięki zapisane w pamięci Spectrum nie są rewelacyjne (w porównaniu np. z AMIGĄ), lecz są zaskakująco dobre jak na komputerek tej klasy, więc ...

Życzę powodzenia przy zabawie z samplingami.

Maciej BRØMBA Pietraś

```
10
20  ;  SYNTHSET (C) BROMBA'89
30
40 *C-
50 *D+
60
70
80          ORG   60000
90
100         DI
110         LD    BC,65533
120         LD    A,9
130         OUT   (C),A
140         LD    BC,49149
150 GLOSNO  LD    A,14
160         OUT   (C),A
170         LD    BC,65533
180         LD    A,3
190         OUT   (C),A
200         LD    BC,49149
210         LD    A,0
220         OUT   (C),A
230         LD    BC,65533
240         LD    A,2
250         OUT   (C),A
260         LD    BC,49149
270         LD    A,0
280         OUT   (C),A
290
300 POCZAT  LD    HL,00000
310 DLUGOS  LD    DE,00000
320         LD    B,8
330 DELAY1  LD    C,13
340 LOOP1   DEC   C
350         JR    NZ,LOOP1
360         RLC   (HL)
370         JR    C,NOPE
380         PUSH  BC
390         LD    BC,65533
400         LD    A,7
410         OUT   (C),A
420         LD    BC,49149
430         LD    A,%11111101
440         OUT   (C),A
450         JR    NEXT
460 NOPE    PUSH  BC
470         LD    BC,65535
480         LD    A,7
490         OUT   (C),A
500         LD    BC,49149
510         LD    A,255
520         OUT   (C),A
530 NEXT    POP   BC
540         DJNZ  DELAY1
550         INC   HL
560         DEC   DE
570         LD    B,8
580 DELAY2  LD    C,11
590         LD    A,D
600         OR    E
610         JR    NZ,LOOP1
620         EI
630         RET
640
650 END     NOP
```

LISTING 1

Artykuł ten jest kontynuacją opisu programu do tworzenia samplingów dla interface'u AY, publikowanego w poprzednim numerze „Bajtka"

```
10  ; FAST SAMPLING FOR AY
20
30
40          ORG   60000
50
60 LEN      EQU   16384
70 POCZ     EQU   32768
80
90          LD    BC,65533
100         LD    A,7
110         OUT   (C),A
120         LD    BC,49149
130         LD    A,255
140         OUT   (C),A
150
160         DI
170         LD    HL,POCZ
180         LD    BC,65533
190         LD    A,#08
200         OUT   (C),A
210         LD    B,#BF
220         LD    DE,LEN
230 MAIN    LD    A,(DELAY)
240 WAIT1   DEC   A
250         JR    NZ,WAIT1
260         RLD
270         OUT   (C),A
280         RRD
290         LD    A,(DELAY)
300 WAIT2   DEC   A
310         JR    NZ,WAIT2
320         RRD
330         OUT   (C),A
340         RLD
350         INC   HL
360         DEC   DE
370         LD    A,D
380         OR    E
390         JR    NZ,MAIN
400         EI
410         RET
420 DELAY   DEFB  10
430         NOP
```

LISTING 2

CENTRONICS DLA FDD 3000 (CP/M)

Prezentujemy procedury obsługi złącza równoległego w systemie CP/M 2.2 dla stacji FDD 3000 podłączonej do komputera ZX Spectrum z interfejsem „Microface". Ideą opisywanego rozwiązania jest przypisanie pliku systemowego LST do urządzenia fizycznego, jakim jest drukarka typu Centronics. Dzięki temu wszystkie programy CP/M'u (Wordstar, dBase, Turbo Pascal itp.) mogą w naturalny sposób współpracować z drukarką wspomnianego standardu.

(Red.)

Użytkownicy stacji dysków FDD 3000 napotykają na duże trudności w przypadku, gdy dysponując drukarką z interfejsem Centronics, chcieliby ją wykorzystywać do pracy w systemie CP/M. Producent stacji przewidział bowiem współpracę jedynie z drukarką wyposażoną w interfejs RS 232. Drukarki takie stanowią jednak zdecydowaną mniejszość na rynku, poza tym, jeśli użytkownik ma już drukarkę z interfejsem Centronics, to naturalnym dążeniem jest chęć jej pełnego wykorzystania.

Przy pracy w systemie CP/M komputer ZX Spectrum stanowi jedynie stację wprowadzania danych z klawiatury i wyświetlania obrazu, zaś funkcję podstawową przejmuje procesor znajdujący się w stacji. Zatem samo podłączenie drukarki do komputera nie pozwoli jeszcze drukować. System CP/M będzie mógł obsługiwać tak podłączoną drukarkę, jeśli napiszemy specjalny program, który będzie przesyłał dane do drukowania ze stacji do komputera. Sterowaniem drukarką może zająć się komputer ZX Spectrum, gdyż czynność ta nie jest czasochłonna.

Transmisję danych ze stacji do komputera można zrealizować w prosty sposób, jeśli dysponujemy terminalem firmy Polbrit, który ma wbudowane procedury przesyłania danych pomiędzy stacją a komputerem. Przykładowo, wysłanie z poziomu CP/M do terminala sekwencji 6 bajtów: 27,78,AL,AH,LL,LH spowoduje, że następne LL+256*LH bajty nie zostaną wypisane na ekranie lecz zostaną umieszczone w pamięci komputera pod adresem AL+256*AH.

Istnieje oczywiście odwrotna sekwencja sterująca do przesyłania danych z komputera do stacji. Ma ona postać: 27,77,AL,AH,LL,LH. Po wysłaniu tej sekwencji wczytane z terminala LL+256*LH bajty będą pochodziły z pamięci komputera od adresu AL+256*AH. Dodatkowo, wysłanie sekwencji 27,90 powoduje zakończenie wykonywania przez komputer programu terminala i przejście do BASICa.

Jeśli po instrukcji uruchamiającej program terminala umieścimy wywołanie własnej procedury, to wysłanie wspomnianego ciągu bajtów spowoduje jej uruchomienie. Programista w systemie CP/M może zatem wykorzystać 30 KB wolnej pamięci w komputerze np. do przechowywania danych, na niewielki RAMdysk itp.

Aby komputer mógł sterować drukarką należy napisać dwa programy — jeden pod systemem CP/M, który modyfikując BIOS przechwyci zwykłą obsługę drukarki z interfejsem RS 232 i zajmie się wysyłaniem danych do komputera, i drugi dla ZX Spectrum, który te dane skieruje do drukarki.

Omówienie działania programów

Program CENT.ASM (Listing 1) modyfikuje BIOS zastępując standardowe procedury LIST i LISTST własną procedurą transmisyjną. Ponieważ oryginalne procedury są bardzo krótkie, program jest umieszczany w drobnych fragmentach na ich miejscu, a znakomita jego część jest lokowana w wolnym obszarze pamięci nad BIOS'em od adresu 65403.

Dzięki takiemu rozwiązaniu TPA (obszar na programy użytkownika) nie zostaje uszczuplone nawet o jeden bajt. Ponadto należy zauważyć, że przesłanie do komputera danych za każdym razem po jednym bajcie byłoby niezwykle nieefektywne, gdyż do każdego bajtu należałoby dodać 6 bajtów sterujących, a i sama transmisja nie jest szybka.

Z tego powodu program tworzy 64-bajtowy bufor. Dane przeznaczone do wysłania są gromadzone w buforze, aż do napotkania znaku końca linii lub przepełnienia bufora. Dopiero wtedy następuje transmisja. Adresem docelowym w komputerze jest bufor drukarki (23296); dodatkowo ciąg znaków jest rozpoczynany zawsze bajtem o wartości 45 (identyfikator) i kończony bajtem o wartości 255 (znacznik końca). Po transmisji linii program wysyła do terminala komendę 27,90, która powoduje przejście do BASICa i wydruk przesłanej linii.

Tekst programu należy wprowadzić za pomocą dowolnego edytora (np. od Turbo Pascala) i skompilować asemblerem Z80 (np. GEN80). Utworzony kod wynikowy typu .COM należy wykonać. Program będzie obecny w pamięci aż do zimnego startu systemu. **Opisany program modyfikuje jedynie oryginalny BIOS firmy TIMEX.**

Drugi program pracujący w komputerze ma za zadanie wydrukować uprzednio przesłaną linię. Opisywane rozwiązanie współpracuje z interfejsem Centronics typu 'Microface' sprzedawanym dawniej przez CSH.

Jeśli Czytelnik ma inny interfejs to współpraca jest oczywiście możliwa, należy jednak zmienić w procedurze drukującej podprogram DRUKUJ na odpowiedni dla własnego interfejsu. Podprogram ten ma za zadanie wysyłać znak z rejestru A na drukarkę (Listing 2).

Jakiekolwiek przekodowywanie znaków charakterystyczne dla BASICa jest niepotrzebne. Działanie pro-

LISTING 1

```
;------------------------------------------------------------
;     Program instalujacy driver centronics drukarki
;            1990 06 23 (c) R.Magdziak
;------------------------------------------------------------
IMPR      EQU #F746         ;adres procedury LIST w BIOS
IMPRST    EQU #F755         ;adres procedury LISTST w BIOS
PRINTER   EQU #F944         ;procedury wywolywane przez
PRINTSTS  EQU #F959         ;LIST i LISTST
FREE      EQU 65403         ;adres wolnego miejsca nad BIOS
CONOUT    EQU #F736         ;adres procedury CONOUT w BIOS

          ORG #100
          LD DE,IMPR        ;driver jest umieszczany
          LD HL,IMPR1       ;fragmentami w BIOS
          LD BC,14
          LDIR
          LD DE,PRINTER
          LD BC,12
          LDIR
          LD DE,FREE
          LD BC,66
          LDIR
          LD DE,PRINTSTS
          LD BC,7
          LDIR
          LD DE,IMPRST
          LD BC,3
          LDIR
          LD DE,STRING      ;adres napisu informacyjnego
          LD BC,#9
          CALL 5
          JP 0              ;koniec pracy
STRING    DEFM " *** DRIVER CENTRONICS INSTALED! ***"
          DEFB 10,13
          DEFM "1990 R.Magdziak"
          DEFB 10,13,"$"

IMPR1     LD A,C            ;przechowaj znak
          LD HL,TAB -1      ;adres liczby znakow w buforze
          LD C,(HL)
          INC (HL)          ;ustalenie nowej wolnej poz.
          LD B,0
          INC HL            ;adres bufora
          ADD HL,BC         ;adres 1 wolnego elementu
          LD (HL),A         ;zapis do bufora
          JP PRINTER        ;skok do dalszej czesci progr.

          CP 13             ;czy koniec linii ?
          JP Z,DRUK         ;drukujemy bufor gdy tak
          LD A,63           ;pojemnosc bufora
          CP C
          JP Z,DRUK         ;drukujemy gdy pelen bufor

          RET

          ORG FREE          ;wolne miejsce nad bios
DRUK      LD HL,TAB -1      ;adres liczby znakow w buforze
          LD C,(HL)         ;wez liczbe znakow do druku
          LD (HL),0         ;bufor czysty !
          INC HL            ;adres bufora
          INC C             ;1 wiecej na znacznik konca
          LD B,0
          PUSH BC
          PUSH HL
          LD HL,#F95D       ;adres dlugosci ciagu znakow
                            ;w ciagu sterujacym terminal
          INC C             ;1 wiecej na komende
          LD (HL),C
          LD HL,PRINTSTS    ;adres ciagu sterujacego
          LD B,7            ;ciag sterujacy ma 7 bajtow
          CALL SEND         ;wyslij do terminala
          POP HL
          POP BC
          DEC C             ;komenda juz wyslana
          LD B,C
          CALL SEND         ;wyslij zawartosc bufora
          LD BC,#FF         ;znacznik konca
          CALL CONOUT
          LD BC,27          ;'GOTO BASIC',funkcja nr 45
          CALL CONOUT
          LD BC,"Z"
          CALL CONOUT
          RET
SEND      LD C,(HL)         ;wez znak do wyslania
          INC HL            ;adres nastepnego
          PUSH HL
          PUSH BC
          LD B,0
          CALL CONOUT       ;wyslij znak na konsole
          POP BC
          POP HL
          DJNZ SEND
          RET
          DEFB 0            ;ilosc znakow w buforze

TAB       DEFB 27,"N"
          DEFW 23296        ;adres docelowy w ZX
          DEFW 0            ;liczba przesylanych znakow
          DEFB 45           ;komenda

          LD A,#FF
          RET               ;drukarka zawsze gotowa !
```

LISTING 2

```
;-------------------------------------------
;DRIVER CENTRONICS DO TERMINALA POLBRIT
;-------------------------------------------
        ORG 27500
        LD HL,23296     ;bufor drukarki
        LD A,(HL)       ;sprawdz identyfikator
        CP 45           ;czy dane do druku?
        RET NZ          ;nie,koniec pracy.
NEXTC   INC HL          ;adres znaku
        LD A,(HL)       ;wez znak do druku
        CP 255          ;czy koniec?
        RET Z           ;tak,koniec pracy
        PUSH HL
        CALL DRUKUJ     ;wydruk znaku
        POP HL
        JR NEXTC        ;drukuj nastepne znaki
;-------------------------------------------
; TYLKO DLA MICROFACE
;-------------------------------------------
DRUKUJ  PUSH AF
BUSY    LD A,#7F
        IN A,(254)      ;test break
        RRA
        JR NC,ERROR     ;nacisnieta spacja
        IN A,(251)
        AND %10000000   ;test czy BUSY
        LD B,A
        IN A,(251)
        AND %10000000
        CP B
        JR NZ,BUSY      ;zaklocenie
        OR A
        JR NZ,BUSY      ;busy,czekaj!
        POP AF
        LD B,80         ;opoznienie
LOOP    DJNZ LOOP
        OUT (251),A     ;wyslanie znaku
        LD B,255        ;opoznienie
LOOP2   DJNZ LOOP2
        RET
ERROR   POP AF          ;przerwanie druku
        POP AF
        POP AF
        RET
```

LISTING 3

```
  10 RANDOMIZE USR 27500: REM WYDRUK LINII
  20 RANDOMIZE USR 58859: REM RESTART CP/M
  30 GO TO 10
9000 BORDER 0: INK 7: PAPER 0
9010 CLEAR 27499: LOAD *"CENT.BIN"CODE 27500
9020 LOAD *"CPM.C2"CODE : REM KOD TERMINALA
9030 RANDOMIZE USR 58856: REM START CP/M
9040 GO TO 10
```

gramu jest bardzo proste: sekwencja 27,90 przesłana z systemu CP/M do terminala powoduje przejście do BASIC-a. Natychmiast po tym następuje wydruk linii i powrót do terminala.

Aby otrzymać działający program trzeba wprowadzić tekst procedury drukującej przy pomocy asemblera GENS3 i ją skompilować. Utworzony kod wynikowy trzeba zapisać na dysk pod nazwą CENT.BIN. Następnie należy wprowadzić tekst nowego loadera ładującego program terminala (Listing 3) i zapisać go na dysk z autostartem od linii 9000 — najwygodniej pod nazwą START.

Robert Magdziak

Literatura:
[1]-R. Świniarski, System operacyjny CP/M, WNT Warszawa 1988.
[2]-PZ Polbrit, Instrukcja obsługi systemu operacyjnego CP/M ver. 2.2.
[3]-Timex Ltd., Listing źródłowy modułu BIOS (plik CBIOS.MAC).

KOPIOWANIE PLIKÓW
przy użyciu jednego napędu

Posiadacze stacji dysków FDD 3000 firmy Timex, wyposażonej w jedną tylko kieszeń napędu 3", zetknęli się z pewnością z problemem przenoszenia zbiorów między dyskietkami.

Zarówno w systemie TOS, jak i CP/M brak jest programów firmowych umożliwiających takie kopiowanie. Częściowym rozwiązaniem są programy BackUp (TOS) i DLCOPY (CP/M), które pozwalają na powielenie całego dysku (lub wybranych ścieżek — DLCOPY).

Dotychczas prezentowaliśmy w Bajtku program kopiujący pliki w TOS'ie przy użyciu jednego napędu. Obecnie kolej ma podobny program w systemie CP/M.

Omawiany program został napisany w Turbo Pascalu i, ze względu na prostotę i elegancję implementacji operacji dyskowych w tym języku, jest on dość krótką i przejrzystą ilustracją użycia plików „beztypowych" (ang. untyped files) oraz instrukcji BlockRead i Block Write. Istotnym elementem programu jest procedura ResetDisk System, pozwalająca na zmianę dyskietki w trakcie kopiowania.

Opisywany program, nie pretendując do pełnej ogólności, posiada następujące ograniczenia:
— kopiowanie tylko jednego pliku,
— maksymalna długość pliku 44800 bajtów.

Dokonanie wszelkich ulepszeń pozostawiamy jednak Czytelnikowi.

Jonasz Mayer

```
Program SingleDriveCopy;
{*******************************************}
{                                           }
{              (C) JM 1990                  }
{                                           }
{ Program umozliwia kopiowanie jednego zbioru }
{ o max dl. 44 800 bajtow miedzy dyskietkami  }
{ przy jednym napedzie.                     }
{                                           }
{*******************************************}
const Recsize=128;
      Bufsize=350;   { rozmiar bufora 44 800 b }
var
  f,g      : file;
  fname,
  gname    : string [14];
  Buf      : array [1..Recsize,1..Bufsize] of byte;
  RecsRead : integer;

procedure ResetDiskSystem;
begin
  BDOS ($0D);
end;

{-------------------------------------------}

begin
  clrscr;
  writeln('(C) JM 1990  Single Drive Copy');
  writeln;
  write ('file name:');
  readln (fname);
  writeln;
  writeln('Insert source disk and press key');
  repeat until keypressed;
  ResetDisksystem;

  Assign(f,fname);   (*$I-*)
  reset (f);         (*$I+*)
  if IOresult<>0
  then begin
         writeln (fname,' file not exists'); halt;
       end;
  BlockRead (f,buf,bufsize,Recsread);
  close (f);

  writeln;
  writeln ('Insert destiny disk and press key');
  repeat until keypressed;
  ResetDisksystem;

  Assign (f,fname);   {$i-}
  reset  (f);         {$I+}
  if IOresult=0
  then begin
         writeln ('File already exists, aborted.');
         halt;
       end;
  rewrite (f);
  BlockWrite (f,buf,Recsread);
  close (f);
  writeln  ('OK, end of copy');

end.

{*******************************************}
```

SPECTRUMOWA LISTA PRZEBOJÓW

Zainteresowanie naszą Listą znacznie spadło. Powodem mogą być wakacje lub spadek rzeczywistego zainteresowania Spectrum. Liczymy na to pierwsze i na razie Spectrumowa Lista Przebojów trwa!

Grafika:

1. SAVAGE
2. CAPTAIN BLOOD
3. PURPLE SATURN DAY
4. STRIDER
5. HARD DRIVIN

Muzyka 48:

1. CHASE H.Q.
2. SAVAGE
3. TOP GUN
4. TERRORMETER
5. ZATHRAX

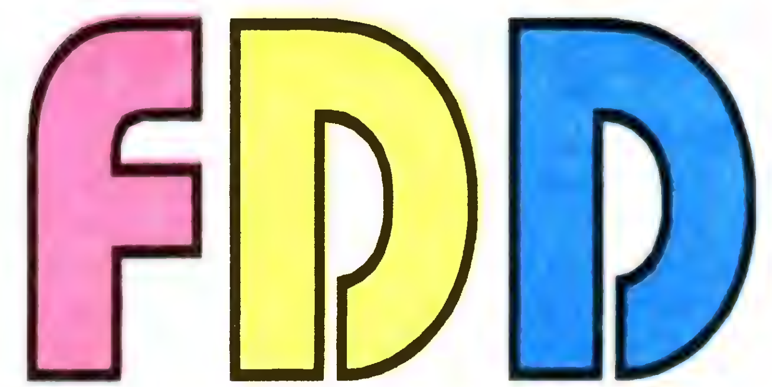
Muzyka 128:

1. UNTOUCHABLES
2. F.I.R.E.
3. PLATOON
4. ROBOCOP
5. GLIDER RIDER

Użytki:

1. SOUND TRACKER
2. TURBO PASCAL 3.0
3. ZEUS
4. ART STUDIO
5. TEXT-ED

5-calowy napęd do



Dziś spełniając listowne prośby Czytelników, przedstawiamy metodykę i podstawy połączenia napędu 5.25" do stacji dysków Timex.

Napędy 5.25" sprawdziły się w eksploatacji jako wygodna pamięć masowa do ZX Spectrum przede wszystkim z uwagi na kilkukrotnie tańsze od dysków 3" dyski 5.25". Koszt zakupu stacji 5.25" zwraca się już przy używaniu 40–50 dyskietek.

Przeróbkę rozpoczynamy od przystosowania naszej stacji do współpracy z dodatkowym napędem. W przypadku stacji FDD 3000 jednokieszeniowej polega to na wyjęciu z wnętrza obudowy stacji dwóch kabli z wtykami: zasilającego i z sygnałami sterującymi. W przypadku dwukieszeniowej stacji FDD 3000 można zrezygnować z jednego napędu 3" lub dobudować zasilacz na napięcia **+5V**, **+12V** i dołączyć nowy napęd jako trzeci.

W celu wyjęcia przewodów z FDD odkręcamy cztery wkręty od spodu obudowy i zsuwamy pokrywę do tyłu. Ponieważ napęd 5.25" nie zmieści się wewnątrz stacji, przewody wyprowadzimy na zewnątrz wyciętą w tym celu szczeliną z tyłu obudowy. Natomiast w przypadku stacji FDD 3, składanej w Polsce, napotykamy na spore trudności, bo wewnątrz stacji nie ma żadnych zapasowych kabelków. Jest rozwiązanie: zasilanie do nowego napędu najłatwiej podłączyć poprzez przylutowanie trzech przewodów (najlepiej zakończonych odpowiednią wtyczką, (rys. 1)) do już istniejącego napędu 3".

Na dołączenie sygnałów sterujących są dwa sposoby: pierwszy — wykorzystać wolną taśmę przewodów sterujących z wtykiem (o ile taka istnieje) lub drugi — kupić wtyk taki jak w napędzie 3" i założyć go na taśmie niżej niż fabryczny o co najmniej 6 cm. Następnie połączyć nowy wtyk z napędem 3", a stary na końcu taśmy z nowym napędem. Może się okazać konieczne sztukowanie taśmy, gdy okaże się ona zbyt krótka.

Następnie trzeba wykonać otwór w górnej części obudowy np. przez usunięcie kilku przegród pomiędzy otworami wentylacyjnymi. Wszystkie prace należy wykonywać bardzo delikatnie i ostrożnie gdyż łatwo uszkodzić stację dysków.

Decydując się na zakup dodatkowego napędu należy zapoznać się z wybranymi charakterystykami napędów 5.25":

— lepiej nie kupować **napędu o mniej niż 40 ścieżkach** (ang. track) np. produkcji węgierskiej, gdyż system TOS źle z nimi współpracuje

— **napęd 40-ścieżkowy jednostronny** np. od bardzo starych komputerów IBM nadaje się, ale aby zapisać drugą stronę dysku trzeba wycinać w dyskietkach dwa otwory: indeksowy i ochrony przed zapisem, co jest bardzo niewygodne, pojemność — 140 kB na stronie

— **napęd 40-ścieżkowy dwustronny** np. taki, jak do IBM XT/AT doskonale współpracuje z systemem TOS, ale do wybierania strony dyskietki trzeba zamontować przełącznik, pojemność — 140 kB na stronie

— **napęd 80-ścieżkowy** dwustronny umożliwia zapis 620 kB na dysku, system TOS obsługuje obie strony jednocześnie, ale niektóre programy kopiujące mogą na nim nie działać

Producentów napędów 5.25" jest wielu, a najlepszą opinią cieszą się firmy: **TEAC, HITACHI, SIEMENS, NEC, BASF.** Ewentualnie można nabyć napęd z NRD lub Węgier, ale należy sprawdzić ilość ścieżek. Podobnie należy upewnić się, czy napęd wyposażono w złącze standardu **SHUGART**, bo inne jest bezużyteczne!

Z uwagi na spory koszt napędów, warto je sprawdzać w momencie zakupu.

Po podłączeniu przewodów zasilającego i z sygnałami sterującymi (uwaga: dolne styki we wtyku są połączone z masą i trzeba je dołączyć tą stroną do masy w napędzie — pomyłka grozi uszkodzeniem stacji (patrz rys. 2) czeka nas zabawa ze zworkami, które odpowiadają za poprawną pracę napędu.

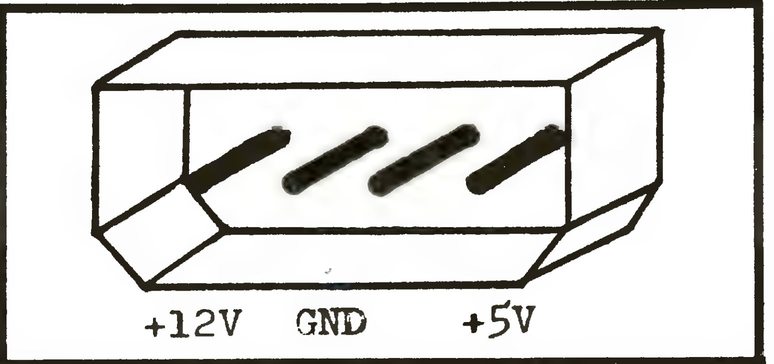
Tuż obok złącza z sygnałami powinien się znajdować w podstawce układ scalony koloru niebieskiego lub czarnego. Jeśli tak, to należy go usunąć. Zabieg ten ma na celu zmniejszenie obciążenia kontrolera dysków. Również obok złącza znajdują się zworki do wyboru napędu. Są one oznaczone **A, B, C, D** albo **0, 1, 2, 3.** Wstawiamy jedną zworkę oznaczając w ten sposób numer napędu np. **B** dla stacji FDD 3 lub **C** dla FDD 3000 dwukieszeniowej.

Teraz po uruchomieniu stacji wkładamy dysk do napędu „A" i dysk 5.25" do nowego napędu i formatujemy dysk 5.25" instrukcją **FORMAT *„B" TO „nazwa"** lub **FORMAT *„C" TO „nazwa"** zależnie od ilości napędów. W przypadku napędu 80-ścieżkowego instrukcję należy zakończyć literą **„D"** np. **FORMAT *„B" TO „dysk_1"D.**

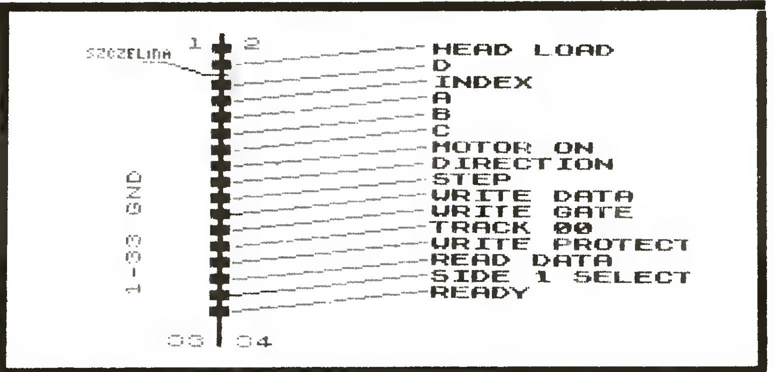
Jeśli dysk został sformatowany poprawnie, to komputer zakończy pracę komunikatem **O.K.** Wtedy spróbujmy podać instrukcję **GOTO *„B"D** i nagrywamy przykładowo **ROM: SAVE *"ROM"CODE 0,16384** a następnie wykonujemy instrukcję **CAT*.** Jeśli plik istnieje, można uznać, że stacja działa prawidłowo. Gdy stacja nie reaguje na polecenia drukując np. komunikat **"DRIVE B DOESN'T EXIST"**, należy zmieniać kolejno ustawienie zworek. Dla napędów dostępnych w redakcji ustawienie zworek podane jest na rysunkach.

UWAGA: W napędzie **A** zawsze powinna znajdować się dyskietka niezależnie od tego, który napęd w danej chwili jest napędem aktywnym!

Na złączu dostępne są sygnały:
— wyjściowe (napęd -> kontroler):



Rys. 1 Gniazdo zasilające

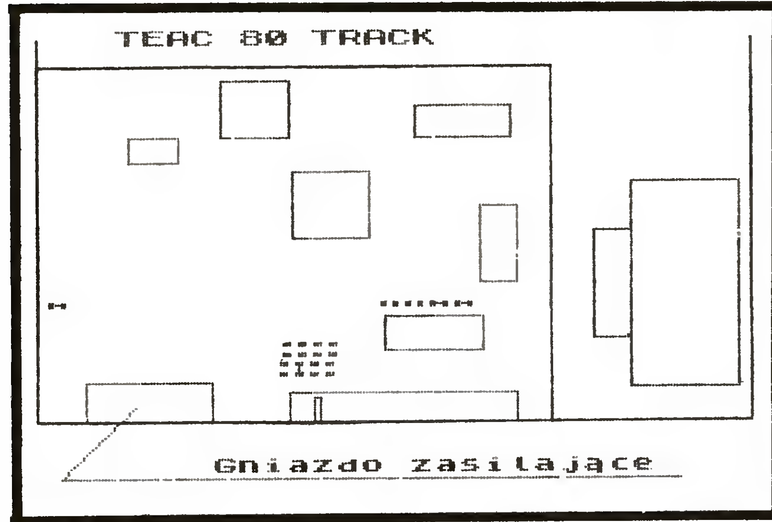


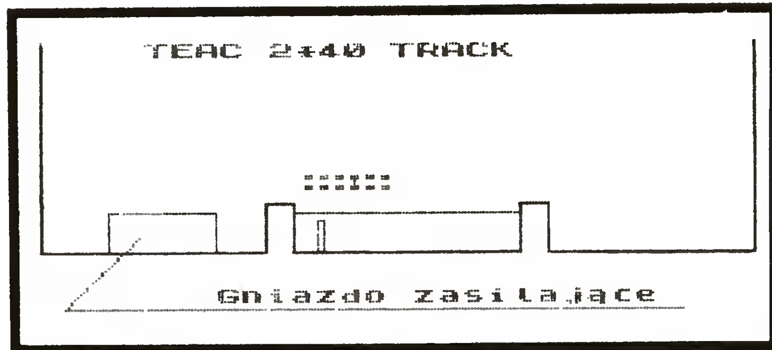
Rys. 2 Złącze z sygnałami

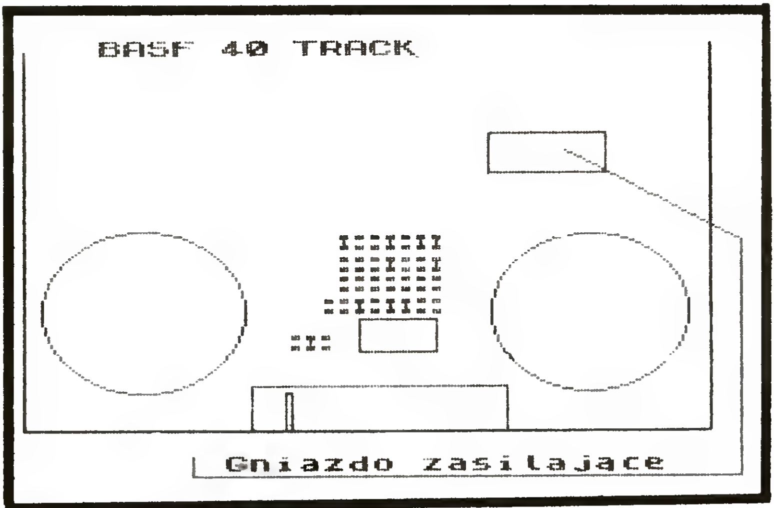
TRACK 00 informuje, że głowica jest na zerowej (zewnętrznej) ścieżce dysku
INDEX sygnalizuje początek ścieżki
READ DATA przesyła szeregowo dane z dysku w czasie odczytu
WRITE PROTECT zabrania kontrolerowi wykonania zapisu lub formatowania dysku (zaklejane wcięcie w dyskietce)
READY sygnalizuje gotowość współpracy
— wejściowe (kontroler -> napęd):
MOTOR ON włącza silnik obracający dyskietkę dla wszystkich napędów naraz
HEAD LOAD wymusza opuszczenie głowicy na powierzchnię dysku (jeśli napęd posiada mechanizm opuszczania głowicy)
STEP impuls powodujący przesunięcie głowicy o jedną ścieżkę
DIRECTION określa kierunek przesuwu głowicy
SIDE 1 SELECT wybiera stronę dysku w przypadku stacji dwustronnych
WRITE GATE (WRITE ENABLE) włącza zapis na dysku
WRITE DATA przesyła szeregowo dane podczas zapisu dysku
DRIVE A, B, C, D wybiera aktywny napęd dyskowy

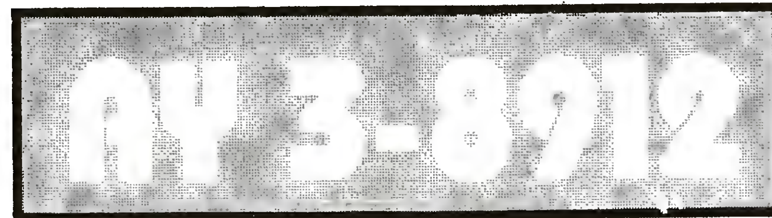
Oprócz tych sygnałów napęd wymaga napięć zasilających **12V — 1A** i **5V — 1A.** Napęd 40-ścieżkowy należy dodatkowo wyposażyć w ręczny przełącznik stron dysku według zamieszczonego schematu. Sygnał **SIDE 1 SELECT** należy odłączyć od gniazda kontrolera np. przecinając ścieżkę, a w to miejsce podłączyć przewód z przełącznika.

Pozostaje teraz problem stale świecącej diody wyboru napędu. Można ją pozostawić bez zmian albo wstawić pro-









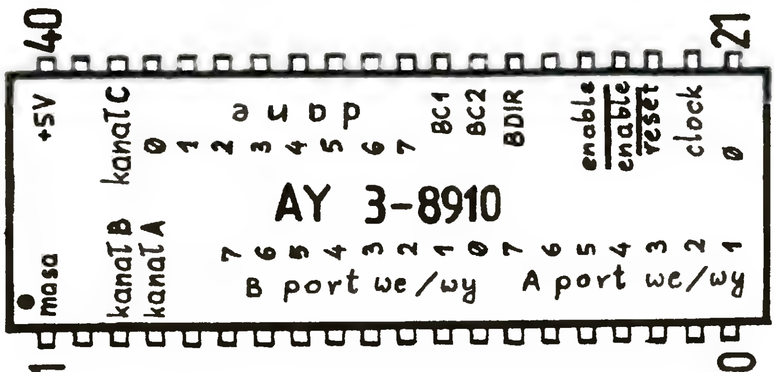
Obiecaliśmy, więc jeszcze raz wracamy do sprawy generatora dźwięku do ZX Spectrum. Tuż po opublikowaniu schematu generatora w numerze 3–4/90 nadeszły listy od osób posiadających układ AY 3-8912 zamiast AY 3-8910 i mających kłopoty w przystosowaniu schematu do tego właśnie układu.

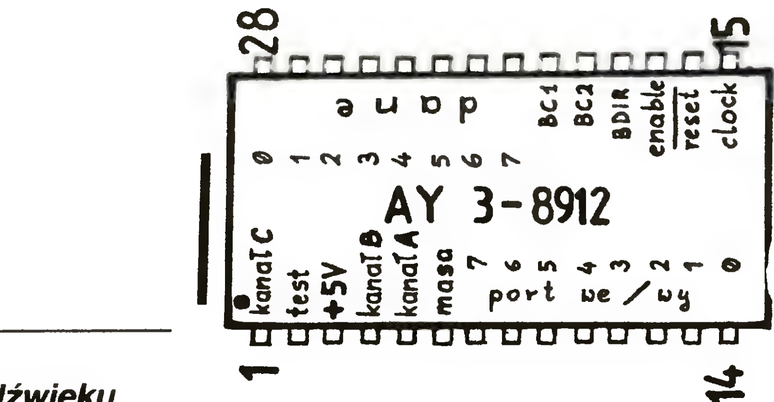
Zamieszczamy więc, na prośbę Czytelników, dokładny opis końcówek obu układów zdając sobie sprawę, że układ AY 3-8912 jest dużo łatwiejszy do zdobycia.

Można się o niego starać w punktach napraw mikrokomputerów jako że jest on zamontowany m.in. w Amstradzie 6128 i ZX Spectrum +2. Z tego samego powodu jest on także łatwy do zdobycia na giełdach części elektronicznych do komputerów. Oba układy AY różnią się między sobą obudową i co za tym idzie funkcjami poszczególnych nóżek. Oprócz tego układ AY 3-8910 ma dwa porty wejścia/wyjścia, a 8912 tylko jeden, więc jest teoretycznie gorszy, ale co ważniejsze tańszy. Gorszy jest na szczęście tylko teoretycznie, bo do naszych potrzeb nadają się równie dobrze oba układy.

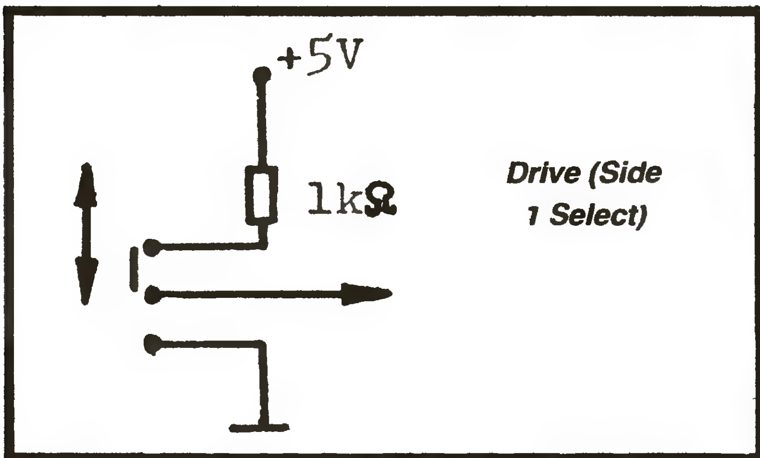
Jak zamontować AY 3-8912 na płytce zaprojektowanej dla 8910? Otóż należy przeciąć połączenia wiodące z punktów 4, 24, 25 i 28 na płytce i wlutować 40-nóżkową podstawkę. Teraz trzeba połączyć drutem:

— przeciętą ścieżkę z punktu 26 z punktem 27,
— punkt 28 z napięciem +5V,
— punkt 22 z punktem 24,
— punkt 23 z punktem 25,
— punkt 4 z punktem 38,





Wyprowadzenie układów scalonych generatorów dźwięku



Wybór strony dla drive'u 2x40 TRACK

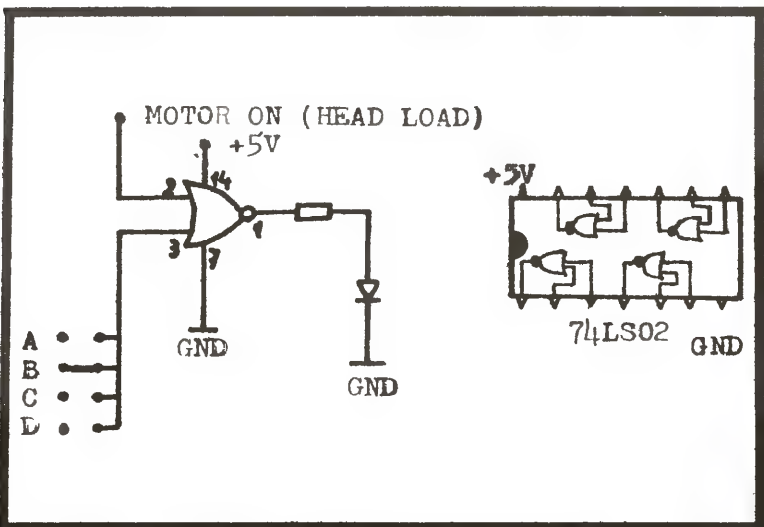
sty układ. Najprostsze będzie podłączenie diody do dwóch sygnałów: **MOTOR ON** i **DRIVE SELECT** z tej strony zworek, z której wszystkie zworki są połączone wspólną ścieżką. Przewód do **MOTOR ON** lutujemy tuż przy złączu uważając, aby nie pocynować ścieżek złącza. Omawiany układ wraz z układem scalonym **7402,** a jeszcze lepiej **74LS02** przedstawiony jest na rys. 4. Układ scalony najlepiej przykleić „plecami" do płytki drukowanej lub obudowy, a zasilanie do niego pobrać bezpośrednio zw złącza zasilającego napęd. Można też zamiast sygnału **MOTOR ON** użyć sygnał **HEAD LOAD,** wtedy dioda będzie pulsować zamiast świecić stale.

W czasie uruchomiania stacji z dodatkowym napędem należy wstawić bezpieczniki rurkowe o prądzie max. **1.6A** zamiast zamontowanych o prądzie **1A.**

Do przeróbki stacji potrzebna jest chociaż minimalna wprawa we władaniu lutownicą, dlatego stanowczo nie polecamy dokonywania przeróbek osobom niedoświadczonym.

Grzegorz Ostapiuk
Tomasz Tarczyński
Marek Sawicki

2 —	1 — GND
4 — HEAD LOAD (INUSE)	3 — GND
SLOT	
6 — DRIVE „D"	5 — GND
8 — INDEX	=
10 — DRIVE „A"	=
12 — DRIVE „B"	
14 — DRIVE „C"	
16 — MOTOR ON	
18 — DIRECTION	
20 — STEP	
22 — WRITE DATA	
24 — WRITE ENABLE (WRITE GATE)	
26 — TRACK 00	
28 — WRITE PROTECT	=
30 — READ DATA	=
32 — SELECT HEAD (SIDE 1 SELECT)	31 — GND
34 — READY	33 — GND



Podłączenie diody

— punkt 6 z napięciem +5V,
— punkt 3 z punktem 7,
— punkt 8 z przeciętą ścieżką z punktu 4,
— punkt 6 z masą zasilania.

Następnie można wlutować wszystkie elementy i na koniec włożyć układ AY 3-8912 tak, aby jego noga nr 1 znalazła się w otworze nr 4 40-nóżkowej podstawki. Wówczas zgadzają się połączenia nóżek 18–28 układu z funkcjami ścieżek na płytce.

To był opis postępowania dla osób, które zdążyły już zrobić płytkę, ale inhym radzę przeprojektować tworzoną wg schematu z „Bajtka" 3–4/90 płytkę na podstawie rad powyżej, aby nie musieli nic łączyć drutem albo zbudować generator na płytce uniwersalnej.

Uwaga dla wykonujących płytkę AY SOUND INTERFACE z nr „Bajtka" 3—4/90: Na schemacie z tego numeru zworkę z punktu 28 układu AY 3-8910 trzeba połączyć z napięciem +5V, zamiast narysowanego połączenia z masą! Pozostawienie starego połączenia powoduje, że układ AY pozostaje nieaktywny!

Marek Sawicki



JAK KUPOWAĆ (TANIE) KOMPUTERY

Przez kilka ostatnich lat asystowałem przy zakupach różnych komputerów kompatybilnych z IBM PC. Kupowany zestaw na ogół składa się z samego komputera, monitora, klawiatury oraz bliżej nieokreślonego zbioru wad, które wyjdą na jaw w ciągu pierwszego miesiąca eksploatacji. Niektóre z tych wad daje się łatwo wykryć podczas odbioru komputera. Głównie im poświęcam ten tekst.

Jeśli już podjąłeś decyzję o kupnie komputera, musisz jeszcze wybrać konkretny model (XT, AT, 386) i konkretną konfigurację. O wyborze modelu decydują głównie przewidywane zastosowania albo zasobność kiesy, co jest tematem godnym osobnego potraktowania. Odłóżmy więc te rozważania na bok, a zajmijmy się wyborem konfiguracji. To, co kupisz, składać się będzie z dwóch części — tego, co niezbędne, i tego, co dodatkowe. Zacznijmy od tego

co musisz kupić.

Na ogół w ofertach handlowych wymieniana jest lista elementów wchodzących w skład proponowanej konfiguracji. W tanich ofertach często brak dość istotnych rzeczy, dlatego też koniecznie sprawdź czy w ofercie znajdują się — pamięć (często jest tylko informacja **„0 kB"**, mówiąca o tym, że za zainstalowanie kości pamięci trzeba zapłacić osobno), monitor, klawiatura (tych trzech rzeczy brakuje najczęściej), zasilacz, obudowa, karta graficzna (np. Hercules), karta Multi I/O (kontrolująca pracę stacji dyskietek, pozwalająca na podłączenie drukarki i komunikację z innymi komputerami, dodatkowo zawiera wbudowany zegar z kalendarzem) i stacje dyskietek. Konieczna też jest gwarancja. Bez tego wszystkiego nie można pracować. Natomiast

co warto mieć?

Z moich obserwacji wynika, że nie warto kupować komputera w skromniejszej konfiguracji niż 640 kB pamięci i dwie stacje dyskietek. Powody są proste — wiele programów odmawia pracy z tylko jedną stacją, a pamięć 512 kB może szybko okazać się za mała — przykładowo, Turbo Pascal 5.0 pracujący pod kontrolą systemu MS DOS 3.30 nie pozwoli uruchomić programu większego niż kilkadziesiąt linii.

Co jeszcze? Przede wszystkim twardy dysk z kontrolerem (czyli sterownikiem, zwykle jest to osobna karta). Mysz, drukarka, koprocesor — ułatwiają pracę, ale korzyści z kupna twardego dysku są znacznie większe — w skrajnych przypadkach Twoja wydajność pracy może wzrosnąć kilkakrotnie. Jeżeli stać Cię tylko na bardzo skromną konfigurację, to pamiętaj, że jeśli nie chcesz rwać za rok lub dwa włosów z głowy, nie powinieneś kupować komputera nie mającego gniazda pod koprocesor, przynajmniej trzech wolnych slotów (czyli gniazd na karty rozszerzające) i miejsca w środku na twardy dysk. Nie radzę też kupować XT z zegarem taktującym szybszym niż 10 MHz — mniejszym kosztem (w finansach i kłopotach) wzrost szybkości można uzyskać zastępując procesor 8088 odpowiadającym mu procesorem NEC V20.

Taki najprostszy zestaw można obecnie (sierpień) kupić w granicach sześciuset dolarów razem z cłem, wystarczy przejrzeć ogłoszenia w kilku gazetach. Po wybraniu oferty i zdecydowaniu się na konkretną konfigurację, a także po odstaniu kilku kolejek w bankach, nadchodzi

dzień odbioru.

Podstawową zasadą odbierania komputerów od sprzedawcy jest to, że jeden z was musi osiwieć. Jeśli nie sprawdzisz dokładnie kupowanego sprzętu przed jego odebraniem — osiwiejesz po tygodniu zmagań z klawiaturą lub stacją dyskietek. Jeśli dokładnie sprawdzisz kupowany komputer, i o wszystkich usterkach powiesz sprzedawcy, on osiwieje natychmiast. Ze społecznego punktu widzenia korzystniejsze jest osiwienie jednego sprzedawcy niż wielu nabywców, dlatego też radzę dobrze przygotować się do odbioru. (Uwaga! Nie wszystkie firmy dają możliwość sprawdzenia sprzętu przy odbiorze, nawet jeśli macie roczną gwarancję, nie radzę kupować kota w worku.) W jaki sposób? Cóż, po pierwsze, istnieją programy testujące komputery, i warto wziąć taki program ze sobą. Dzięki niemu powinno udać się sprawdzić, czy kontaktują wszystkie klawisze w klawiaturze, czy stacje dyskietek odczytują i zapisują informacje w sposób zgodny z oczekiwaniami, a także, jak szybko działa komputer. W przypadku twardego dysku programy testujące są jedynym sposobem uzyskiwania jakichkolwiek informacji na jego temat. Są jednak pewne rzeczy, których nie da się sprawdzić nawet przy pomocy najbardziej wyrafinowanego oprogramowania. W celu upewnienia się, czy komputer nadaje się do pracy, należy poddać go kilku próbom.

Najprostsza z nich polega na wyłączeniu komputera, odczekaniu kilku sekund, i włączeniu go ponownie. Jeżeli w ciągu kilku sekund na ekranie nie pojawi się komunikat o testowaniu pamięci, naciskamy guzik **reset**. Jeśli to pomoże, rezygnujemy z zakupu tego egzemplarza. Zawieszanie się komputera w momencie jego uruchamiania jest rzadkie i niegroźne, ale potrafi doprowadzić do białej gorączki.

Do drugiej próby, którą będzie

próba stacji dyskietek

wybieramy starannie dobrą, już sformatowaną dyskietkę 1.2 MB, najlepiej bez usztywniającego plastikowego pierścienia w środku. Formatujemy ją ponownie w stacji dużej gęstości poleceniem **format/4**, i kopiujemy na nią (ciągle w stacji 1.2 MB) kilka zbiorów (najlepiej system i dowolny program użytkowy). Wkładamy tak przygotowaną dyskietkę do stacji A: kupowanego komputera i resetujemy go, a po wystartowaniu systemu uruchamiamy zapisany na dyskietce program. Dobrze jest całą operację powtórzyć kilka razy, za każdym razem otwierając stację i zamykając ją po lekkim poruszeniu dyskietki. Dodatkowo można (a nawet trzeba) przetestować tą samą dyskietką drugą stację. Marne napędy będą miały kłopoty z odczytywaniem przygotowanej dyskietki, choć powinny być w stanie to zrobić.

Kolejnym

słabym punktem bywają monitory.

Dwie ich podstawowe wady (przynajmniej w przypadku monitorów monochromatycznych) to zniekształcenia

obrazu lub jego nieostrość w pewnych częściach ekranu, i poświata. Pierwsze dwie wady dają się wykryć podczas dokładnych obserwacji ekranu pokrytego siatką prostopadłych linii (coś w rodzaju obrazu kontrolnego w telewizorze, linie nie mogą być za blisko siebie, dla karty Hercules optymalna odległość to około piętnastu pikseli). Poświatę widać przy czyszczeniu ekranu poleceniem **cls**, kiedy znaki już skasowane są jeszcze przez jakiś czas lekko widoczne. Dotyczy to zwłaszcza monitorów białych (paper white), mających czasem zieloną poświatę, bardzo wolno znikającą i męczącą oczy. Niezależnie od tego, czy będziecie dysponować podczas kupna możliwością obejrzenia obrazu kontrolnego na ekranie, koniecznie przełączcie kilka razy tryb tekstowy na graficzny i z powrotem. Im gorszy jest monitor, tym większe będzie przesunięcie obrazu, w granicznych przypadkach może się zdarzyć tak, że po zmianie trybu konieczne jest regulowanie monitora, żeby możliwe było poprawne odczytanie całego ekranu. Taki monitor nie nadaje się do pracy.

Ostatnim elementem, którego sprawdzenie jest proste i możliwe do wykonania przy odbiorze komputera, jest

klawiatura.

Ponieważ z nią ma się najczęściej do czynienia, warto jej poświęcić kilka minut. Oprócz sprawdzenia działania wszystkich klawiszy (o czym była mowa wcześniej), radzę przeprowadzić bardzo prosty test ergonomiczności. Siadamy możliwie wygodnie przed komputerem (dobrze, żeby klawiatura leżała na stole możliwie zbliżonym wysokością do tego, na którym będzie jej stałe miejsce), i piszemy pół strony dowolnego tekstu z maksymalną szybkością jaką potrafimy osiągnąć. Jeśli podczas tego pisania nic nam nie przeszkadza, to dobrze, klawiatura nadaje się do użytku. W przeciwnym razie — cóż, dobrze jest przetestować jeszcze jedną, albo kilka.

Najczęściej kłopoty związane są z odmiennym rozłożeniem klawiszy (zwłaszcza escape i backslash, czyli |), mają tendencje do wędrowania z jednego końca klawiatury na drugi. Nie jest to wielkim problemem, gdyż można się do tego przyzwyczaić. Znacznie gorzej jest wtedy, gdy klawisze naciśnięte pod kątem odbiegającym od pionu blokują się i nie dają żadnego sygnału. Taka klawiatura nie nadaje się do pracy na dłuższą metę, tak samo jak klawiatura „gubiąca" lub „mnożąca" znaki, czy też nie powtarzająca ich przy dłuższym naciskaniu (to ostatnie może być objawem wady nie tylko klawiatury, ale i płyty głównej). Z klawiaturami „cykającymi", czyli wydającymi lekki stuk po naciśnięciu klawisza, na ogół nie ma problemów, ale i one potrafią zachowywać się w sposób nieoczekiwany — niektóre z nich mogą reagować na naciśnięcie zanim rozlegnie się stuknięcie. Łatwo to sprawdzić — wystarczy lekko dotknąć klawisz, nie wciskając go do oporu. Jeśli komputer zareaguje, mimo że klawiatura nie wydała żadnego dźwięku, może się to zdarzyć również w czasie pracy. Nie dyskwalifikuje to wprawdzie klawiatury, ale czasami może okazać się uciążliwe.

Dobrym sprawdzianem całości jest uruchomienie jakiegoś programu wymagającego możliwie dużej zgodności komputera ze standardem IBM. Bardzo czułe na drobne nawet braki w kompatybilności są między innymi takie programy Microsoftu jak Flight Simulator czy edytor Word (nie mylić z edytorami WordStar i WordPerfect). Ostatecznym testem powinno być uruchomienie na kupowanym komputerze programu, który będzie używany najczęściej, i wykonanie kilku podstawowych dla niego operacji.

Komputer, który przeszedł przez wszystkie próby, nie powinien sprawiać w pracy większych kłopotów. Oczywiście

nie sposób sprawdzić wszystkiego

— możecie mieć spóźniający się zegar, uszkodzone któreś z łączy karty Multi I/O, napięcie 110 V na obudowie (to nie żart, osobiście mierzyłem), źle zmontowaną obudowę (słabo dokręcone śruby lub płyta główna przesunięta względem obudowy — widać to od tyłu, gniazda kart rozszerzających nie są ustawione równolegle do krawędzi szczelin w obudowie, czasem powoduje to kłopoty przy dołączaniu drukarki lub myszy). Osobny temat to długość kabli przy monitorze — są one bardzo często za krótkie na to, by monitora można było używać przy komputerze w obudowie typu **tower**, stojącej na podłodze. Znam trzy komputery 386 stojące z tego powodu na drewnianych skrzyneczkach wysokości około trzydziestu centymetrów. Fachowcy przywieźli i postawili komputery na stole, pokazali, że działają, a potem poszli, mówiąc — „Ustawcie je sobie Państwo tak, jak Wam wygodnie". Nie dajcie się na to nabrać.

Mam nadzieję, że przy pomocy tego tekstu przynajmniej częściowo uda Wam się ustrzec kłopotów podczas zakupu własnego komputera. Jeżeli nie — będę wdzięczny za uwagi na temat innych wad wrodzonych kupowanego przez Was sprzętu.

Tekst ten powstał przy intensywnej, choć nieświadomej współpracy kilku warszawskich firm komputerowych, które chyba nie byłyby zbyt zadowolone, gdybym wymienił ich nazwy.

Marcin Borkowski



Niektóre komputery AT mają podczas ich uruchamiania zwyczaj przełączania klawiatur typu PS/2 (z klawiszami funkcyjnymi u góry) w tryb NumLock.

Mimo iż ma to swoje logiczne uzasadnienie w możliwości pełnego wykorzystania wszystkich klawiszy nowych klawiatur, wielu użytkowników PC zaczyna pracę od przekleństw i łupnięcia w klawisz NumLock, aby móc pracować zgodnie ze starymi przyzwyczajeniami. Na ich usprawiedliwienie można powiedzieć, że niektóre programy wymuszają takie działania, gdyż klawisze **Del** z klawiatury numerycznej i **Delete** z bloku sterowania kursorem nie są dla nich równocenne (tak jest na przykład w przypadku edytora Word). Istnieje prosty, i wypróbowany sposób oszczędzenia klawiatury i nerwów użytkownika — wystarczy w autoexec-u umieścić program, który zmieni stan klawiatury. Trzeba w tym celu wyzerować piąty bit bajtu spod adresu 0:0417H. Robiący to program powinien być możliwie krótki i szybki, najlepiej więc napisać go w asemblerze. Może wyglądać na przykład tak:

numlock. asm:

```
ASSUME cs: code
code SEGMENT
start:
 xor ax, ax
 mov ds, ax
 and [byte ptr 0:0417h],11011111b
 mov ax, 4C00h
 int 21h
code ENDS
END start
```

Tym, którzy potrafią go tylko wstukać, i nie wiedzą co trzeba zrobić dalej, proponuję jeszcze jeden program, tym razem wsadowy, który wykona za nas całą brudną robotę:

commake. bat:

```
tasm %1
tlink %1
exe2bin %1
if exist %1.com del %1.com
rename %1.bin %1. com
del %1.exe
del %1.map
del %1.obj
```

Po wstukaniu obu wydruków (na przykład pod nazwami **numlock.asm** i **commake.bat**), piszemy **commake numlock** i naciskamy enter. Efekt będzie zależał od tego, czy na dysku znajdują się trzy potrzebne programy: **tasm**, **tlink** (asembler i linker Borlanda) i **exe2bin** (część MS DOS-u). Jeżeli któregoś z nich zabraknie, trzeba będzie dokonać kompilacji u kogoś lepiej wyposażonegpo. Otrzymamy program **numlock.com** o długości czternastu bajtów. Umieszczamy go w autoexec-u w dowolnym miejscu, i od tej pory przestajemy zwracać uwagę na fochy klawiatury.

Program **commake.bat** może być przydatny (i wygodny) za każdym razem, kiedy chcemy z zapisanego w asemblerze programu uzyskać zbiór typu ***.com**, dlatego warto go zostawić na dysku, i nie kasować.

Marcin Borkowski

LEMONIADA

Pisałem poprzednio o wspaniałości i doskonałości gier na IBM. Dla kontrastu opisze teraz grę, która składa się tylko z pomysłu. A imię jej LEMON.

Tekst źródłowy gry napisany został zapewne w Turbo Pascalu i tako skompilowany. Cała gra mieści się więc w malutkim pliku .COM. Wykorzystuje tryb tekstowy, więc nie będzie problemów z konfiguracją.

Gra polega na popularnej ostatnio zabawie w handlarza ulicznego. Tym razem jest pełnia lata, słoneczny kurort, tłum ludzi. Ty rozstawiasz w kącie plaży kramik z lemoniadą. Sezon trwa dziesięć tygodni, to całe Twoje pole do popisu.

Masz 15 dolarów. Kupujesz cukier, lemoniadę w proszku i kubeczki. Liczbę wyprodukowanych jednostek konsumpcyjnych ogranicza surowiec zakupiony w najmniejszej ilości.

Co tydzień kupujesz gazetę, z której dowiadujesz się o pogodzie planowanej na następny tydzień. Według tego możesz ustalać cenę kubeczka lemoniady. Zakupy robisz też raz w tygodniu.

Codziennie spragnieni plażowicze wypijają część Twoich zapasów. Danego tygodnia nie możesz zmienić ceny lub dokupić produktów, jeśli więc okaże się, że leje deszcz, a Ty żądasz 99 centów za kubek, to Twoja strata. Gdy zaś lemoniady zabraknie, zamykasz kramik i czekasz z utęsknieniem końca tygodnia.

I to wszystko. Zagraj w tę grę kilkaset razy, to powinno wyrobić Ci pojęcie o funkcjonowaniu mini-rynku. Pamiętaj też, że gdy sezon się skończy, to nikt nie odkupi od Ciebie ewentualnie nagromadzonych zapasów. Lecz jeśli zagarnąłeś 500 dolarów, to możesz spokojnie oddać cały kram na złom.

Marcin Przasnyski

POKEr

(gra znaczonymi kartami)

Skoro już ustaliliśmy miesiąc temu, że na PC można grać, pójdźmy krok dalej — zacznijmy grzebać w grach. Tu POKE, tam POKE, zamienić DEC AX na NOP, i wygrać w cuglach — czy można marzyć o czymś więcej?

W dodatku IBM jest taki wygodny — uruchamia się porządny debugger (może być Borlanda) i wio — niektóre opcje chyba były pomyślane wyłącznie dla poprawiaczy niepotrzebnie trudnych gier. Chwilowo wprawdzie nie mogę wiele zrobić, bo do rozbrajania gier działających w trybie graficznym potrzebne są dwa komputery (śledzi się program z zewnątrz, korzystając z RS 232) a dysponuję tylko jednym, ale taki TETRIS w wersji oryginalnej pracuje w trybie tekstowym, więc **td tetris.com,** kilkanaście minut prób, a potem bierzemy NortonUtilities albo PCTools, i POKE:

0000:47B4 — zamienić 03 na dowolną większą. Pod tym adresem zapamiętana jest ilość punktów, jakie dostaje się za każdy klocek.

0000:4A0A — zamienić 03 na 00, i gra będzie się toczyć cały czas na tym samym poziomie.

Teraz niestraszne nam szybko spadające klocki, bo ich nie ma. A każdy (nawet ten najbardziej wyśrubowany) wynik redaktora Zachara można pobić w ciągu kwadransa. I dobrze mu tak.

Marek Ciężarek

HERKULES na dysku

Niezależnie od skojarzeń, jakie wywołuje tytuł, nie będziemy zajmować się grecką mitologią ani sportem. Od czasu do czasu każdemu, kto używa komputera do wykonywania obliczeń, zdarza się przygotować na ekranie coś, co warte jest uwiecznienia. O ile do dyspozycji mamy kartę Hercules, i pamiętaliśmy o wczytaniu hgcprint-a albo innego programu kopiującego ekran w trybie graficznym, możemy spróbować uzyskać kopię na papierze.

Często jednak jedna kopia nas nie urządza, a że trudno z góry przewidzieć, ile egzemplarzy będzie potrzebnych, jesteśmy w kropce. Żeby tak można było ekran zapisać na dysku...

Można. To, co widać na ekranie, ma swój odpowiednik w postaci pamięci ekranu — obszaru RAM, tak zorganizowanego, że odpowiednie układy potrafią przetłumaczyć każdy bit w nim zapamiętany na kropkę (widoczną lub nie) na ekranie. Wystarczy zapisać zawartość pamięci ekranu na dysku, a potem w odpowiednim momencie wczytać ją z powrotem na swoje miejsce. Przy użyciu Turbo Pascala, który pozwala na umieszczenie zmiennych w dowolnym miejscu pamięci przy pomocy deklaracji **absolute**, sprawa wydaje się być banalna.

Niestety, recepta ta jest tyle samo warta, co propozycja ręcznego odrysowywania zawartości ekranu. Prawdopodobnie jest to związane z konfliktem, jaki pojawia się między różnymi układami z których zbudowany jest IBM, kiedy próbujemy nagrać pamięć ekranu bezpośrednio na dysk. To, co się na nim znajdzie, to nic nie warte śmieci, mające z naszym rysunkiem tyle wspólnego, co rozlana zupa z obrazem Rubensa.

Na szczęście można ten problem obejść. Trzeba w tym celu zrezygnować z prób **bezpośredniego** zapisania na dysku zawartości pamięci ekranu. Najpierw zawarte w niej informacje należy przenieść w jakieś inne, bezpieczne miejsce, a dopiero potem wysłać je na dysk. Analogicznie (choć w odwrotnej kolejności) należy postępować przy wczytywaniu ekranu.

Obie operacje przeprowadzają procedury **screensave** i **screenload**, tworzące razem moduł **scrsave** (listing 1). Procedury należy wywoływać z jednym parametrem — nazwą przewidzianą dla zbioru zawierającego ekran. Można z nich korzystać w swoich programach w dowolnym momencie (o ile system jest w trybie graficznym), a nagrane na dysk ekrany wygodnie jest oglądać (i ewentualnie drukować) korzystając z pomocy programu **showscr** (listing 2). Dla zbiorów zawierających pamięć ekranu zastosujemy rozszerzenie **scr** — ponieważ będziemy z nich jeszcze w przyszłości korzystać, nie warto szukać własnych, innych rozszerzeń.

Moduł **scrsave** w swojej obecnej postaci (zgodnie z wydrukiem), jest przeznaczony do nagrywania na dysk pierwszej strony graficznej Herculesa. Jeżeli komuś zależy na nagraniu drugiej strony, musi zamienić adres zmiennej **screen** z $B000:$0000 na $B800:$0000. Można również zmodyfikować nieco treść procedur tak, by nagrywaniu ulegała ta strona pamięci, która jest aktywna lub widoczna. Pozostawiam to zainteresowanym.

Marcin Borkowski

LISTING 2 — PROGRAM SHOWSCR

```
{$A+,B-,D-,E-,F-,I-,L-,N-,O-,R-,S-,V-}

unit scrsave;

interface

procedure screensave(name : string);
procedure screenload(name : string);

implementation

uses dos;

var
  savedscreen : file;
  screen : array[0..32767]of byte absolute $B000:$0000;

procedure addext(var name : string);
{ - - - - - - - - - - - - - - - - - - - - - -
  Dodaje do nazwy zbioru rozszerzenie .scr
  - - - - - - - - - - - - - - - - - - - - - - }
var
  p,n,e   :  string;
begin
  fsplit(name,p,n,e);
  name:=p+n+'.scr'
end;

procedure screensave(name : string);
{ - - - - - - - - - - - - - - - - - - - - - -
  Nagrywa zawartość ekranu na dysk pod
  nazwą name.scr
  - - - - - - - - - - - - - - - - - - - - - - }
var
  i   :  word;
  buf :  array[1..512]of byte;
begin
  addext(name);
  assign(savedscreen,name);
  rewrite(savedscreen,1);
  for i:=0 to 63 do
  begin
    move(screen[512*i],buf,sizeof(buf));
    blockwrite(savedscreen,buf,sizeof(buf))
  end;
  close(savedscreen)
end;

procedure screenload(name : string);
{ - - - - - - - - - - - - - - - - - - - - - -
  Wczytuje z dysku ekran nagrany pod
  nazwą name.scr
  - - - - - - - - - - - - - - - - - - - - - - }
var
  i,readed :   word;
  buf      :   array[1..512]of byte;
begin
  addext(name);
  assign(savedscreen,name);
  reset(savedscreen,1);
  if IOresult<>0 then EXIT;
  for i:=0 to 63 do
  begin
    blockread(savedscreen,buf,sizeof(buf));
    move(buf,screen[512*i],sizeof(buf))
  end;
  close(savedscreen)
end;

end.
```

LISTING 1 — MODUŁ SCRSAVG

```
uses scrsave,crt,graph;

var
  crd,md : integer;

begin
  if paramcount=0 then
  begin
    writeln('Potrzebny parametr - nazwa zbioru...');
    HALT
  end;
  crd:=0;
  initgraph(crd,md,'');
  screenload(paramstr(1));
  repeat until keypressed;
  closegraph
end.
```

Z KOMPUTEREM PRZY ŚWIECACH

czyli o zasilaczach awaryjnych

Zacznijmy od hasła: „Jeśli dane zasługują na przetwarzanie, to zasługują również na zabezpieczanie". Podobne zdanie można zapewne napisać o sprzęcie komputerowym, spróbujmy się więc przyjrzeć zagrożeniom danych i sprzętu, oraz możliwym zabezpieczeniom.

Niewątpliwie ważne jest fizyczne zabezpieczenie przed kradzieżą czy zniszczeniem cennych komputerów, trzeba też chronić przed kradzieżą (skopiowaniem) ważne, poufne dane. Po co bowiem wspaniały system haseł chroniący programy i zbiory, gdy w nocy przyjdzie złodziej, instalację wraz z systemem haseł wsadzi do worka i razem znikną w sinej dali. Jednak jest to temat dla specjalistów z innych dziedzin. Ale bardzo często bez udziału złodzieja tracimy dorobek wielu godzin lub dni pracy.



Nawet trwająca mniej niż sekundę przerwa w dopływie energii elektrycznej, gwałtowny skok napięcia czy też przeniesione przez sieć silne zakłócenie powodują w pracy komputera przerwę, po której cała zawartość pamięci operacyjnej jest stracona. Nie są to wcale zagrożenia wydumane, czysto teoretyczne, nie są też dla większości użytkowników tajemnicą. Dość powszechnym i na pewno godnym polecenia nawykiem jest okresowe zapisywanie dorobku z pamięci operacyjnej do nieulotnej pamięci zewnętrznej. Jeśli np. pisząc tekst artykułu lub programu, co godzinę zapisuję go na dysk, to w najbardziej złośliwym przypadku (awaria tuż przed kolejnym zapisem) będę musiał powtórzyć tylko pracę ostatniej godziny — a nie np. całego dnia. A to dzięki możliwości powrotu do ostatnio zrobionej kopii.

Powyższa metoda jest z powodzeniem stosowana zarówno przez amatorów jak też i przez zawodowców. Niestety, w profesjonalnych zastosowaniach nie zawsze jest ona wystarczająca. Nie zawsze bowiem możemy zgodzić się na jakąkolwiek, choćby najkrótszą przerwą w pracy. Przyczyn może być kilka: odtworzenie utraconych danych może być bardzo pracochłonne lub nawet niewykonalne, np. jeśli rejestrujemy na bieżąco przebieg jakiegoś zjawiska fizycznego. Nie możemy również przerwać pracy w tzw. czasie rzeczywistym, np. jeśli komputer steruje ruchem czy obsługuje salę operacyjną. W takich sytuacjach, oprócz zapewnienia niezawodności sprzętu i oprogramowania musimy postarać się o niezawodne dostawy energii elektrycznej.

Zwykle energii dla naszych komputerów dostarcza publiczna sieć elektryczna. Niestety, często dostarcza ona nie tylko energii, ale również sporo problemów. Najczęściej spotykane to: okresowy spadek napięcia poniżej dopuszczalnej normami wartości (analogicznie wzrost ponad dopuszczalną wartość), impulsy o bardzo wysokim napięciu, zakłócenia skrótowo określane jako „trzaski", i wreszcie chyba najpoważniejsze, całkowite wyłączenie. Znaczna część tych plag wydaje się być trudna lub prawie niemożliwa do wyeliminowania. Np. impulsy mogą być spowodowane uderzeniem pioruna w naziemne urządzenia energetyczne. Dodatkowo, takie uderzenie może spowodować zadziałanie automatycznych bezpieczników w sieci, i krótkotrwałe wyłączenie. Sieć podlega zmiennym obciążeniom, spowodowanym indywidualnymi włączeniami i wyłączeniami urządzeń u tysięcy niezależnych użytkowników. W każdym otoczeniu znajdują się odbiorniki dużej mocy (np. silnik windy), których włączanie stanowi poważne obciążenie. Przykłady można by mnożyć, ale i bez tego chyba wszyscy się zgadzamy, że na napięciu sieciowym nie można zbytnio polegać. I nie jest to wcale problem tylko Polski. Wyniki badań sieci elektrycznych, przeprowadzonych przez firmę IBM wykazały występowanie miesięcznie przeciętnie prawie 129 zakłóceń.

Sporo powiedzieliśmy o stratach danych spowodowanych awariami sieci, ale przecież duże zakłócenia sieciowe powodują również uszkodzenia sprzętu, co oczywiście pociąga dalsze straty — koszty napraw, przestojów, zmniejszenie żywotności sprzętu. Różne składniki systemów komputerowych nie są jednakowo wrażliwe na awarie.

Kosztem pośrednim awarii sprzętu i/lub utraty danych jest zmniejszenie się zaufania użytkowników i zniechęcenie do pracy z systemami informatycznymi.

Wrażliwość komputerów na zakłócenia zasilania nie jest zjawiskiem nowym. Od dawna też próbowano zaradzić złu. Jak? Zasadnicza odpowiedź jest bardzo prosta: skoro zasilanie prosto z gniazdka jest takie niezdrowe, to trzeba je przed podaniem komputerowi uzdatnić. Warto może dodać, że kłopoty z zasilaniem dotyczą nie tylko komputerów, ale właściwie wszystkich skomplikowanych i delikatnych urządzeń elektronicznych.

Dawno, dawno temu życie było prostsze. Duże komputery (innych nie było) stały w dużych ośrodkach i problem zasilania można było rozwiązać generalnie. Powszechnym rozwiązaniem była przetwornica. Składała się ona z silnika elektrycznego oraz prądnicy, osadzonych na wspólnym wale. Silnik włączony do sieci obracał się i napędzał prądnicę, która zasilała ośrodek obliczeniowy. Nie była to wcale kolejna próba zbudowania perpetuum mobile, lecz poważne urządzenie stabilizujące. Dzięki dużej bezwładności mechanicznej taki układ eliminował wszystkie drobne niestabilności i zakłócenia.

Przetwornica nie zabezpiecza jednak przed wyłączeniem prądu. Jeśli chcemy być zupełnie niezależni, to musimy mieć własne zasoby energii elektrycznej, czyli naładowane akumulatory, lub własne, dodatkowe źródło prądu, np. spalinowy agregat prądotwórczy, kawałki bursztynu do pocierania czy też akwarium z węgorzami elektrycznymi.

Mimo burzliwego rozwoju mikrokomputerów duże, wydzielone ośrodki obliczeniowe nadal istnieją i szczęśliwie rozwiązują swoje problemy, my jednak ograniczymy się do warunków i urządzeń, które możemy spotkać w domu czy biurze. Czy w takich warunkach w ogóle potrzebne są zabezpieczenia? Czy wspomniane wcześniej kopie zapasowe nie są wystarczające?

W wielu prostych zastosowaniach wystarczą. Ale pojawienie się wielozadaniowych i wielodostępnych systemów operacyjnych dla mikrokomputerów profesjonalnych znacznie komplikuje sytuację. Jeśli komputer pracuje np. pod systemem UNIX, i wykonuje zadania kilku użytkowników, korzystając przy tym z kilkudziesięciu zbiorów dyskowych, to skutki nagłego przerwania pracy wcale nie są proste do usunięcia. Podobnie rzecz się ma z sieciami lokalnymi. Wyróżniony mikrokomputer (serwer) pełni funkcje usługowe wobec wszystkich pozostałych użytkowników (stacji roboczych). Przerwanie pracy serwera w trakcie obsługi przesłań dyskowych to nie tylko unieruchomienie sieci, ale dezorganizacja danych wszystkich pracujących użytkowników stacji.

Pamiętajmy wreszcie, że nawet najprostszy, używany do gier komputer, choć nie straci cennych danych (najwyżej Krwawy Bill straci Krwawą Mary), może zostać uszkodzony skutkiem zakłóceń sieciowych.

Istnieje więc potrzeba stosowania zabezpieczeń, mamy więc wiele urządzeń zabezpieczających. Najprostsze (i najtańsze) to zwykłe filtry, pozwalające wyeliminować zakłócenia elektromagnetyczne, atmosferyczne i wysokiej częstotliwości przenoszone przez sieć. Nie zabezpieczają one jednak przed zmianami napięcia.

Pełną filtrację zakłóceń i możliwość regulowania napięcia (a więc również podwyższenia za niskiego napięcia) dają bardziej rozbudowane układy. Są one zbudowane w oparciu o transformatory lub o elektroniczne układy zawierające prostownik i przetwornik. Prostownik zamienia prąd zmienny z sieci na prąd stały, zaś przetwornik przetwarza prąd stały na prąd zmienny o stabilnych parametrach i nie zawierający zakłóceń.

Jeśli chcemy zabezpieczyć się przed brakiem zasilania, to powyższy układ musimy uzupełnić o akumulatory, ładowane w normalnych warunkach z prostownika do pełnej pojemności. W przypadku zaniku napięcia w sieci, urządzenie musi natychmiast zacząć pobierać prąd z akumulatora i poprzez przetwornik dostarczać zasilanie do komputera. Natychmiast, to znaczy w niewielkim ułamku sekundy, tak aby zasilacz komputera nie zauważył przerwy — inaczej wszystko na nic.

Opisane urządzenie po angielsku nazywa się UPS (Uninterruptable Power Supply), po polsku mówimy zwykle zasilacz awaryjny lub zasilacz podtrzymujący napięcie. W języku polskoinformatycznym najczęściej używany termin to ju-pi-es

Warto podkreślić, że prąd stały, czy to z prostownika czy z akumulatora, jest z powrotem przetwarzany na prąd zmienny. Komputer „wewnątrz" jest zasilany prądem stałym, ale odpowiednich napięć dostarcza mu własny zasilacz. Nie stosuje się żadnych obejść tego zasilacza — pobiera on zawsze prąd zmienny i nawet „nie wie" czy jest to prąd bezpośrednio z sieci, czy pośrednio z akumulatora.

Tak samo jak zabezpieczane komputery, zasilacze muszą pracować w pomieszczeniach biurowych czy mieszkalnych. Oznacza to, że nie mogą wydzielać szkodliwych gazów czy wytwarzać dużego hałasu. Oznacza to również, że muszą obyć się bez stałego dozoru serwisowego.

Oferowane na rynku urządzenia z powodzeniem spełniają powyższe wymagania. Akumulatory zwykle są szczelnie zamknięte i nie wymagają żadnej obsługi. Są one budowane tak, aby podtrzymać pracę podłączonych urządzeń przez okres od kilku do kilkudziesięciu minut. Jest to więc zasilanie doraźne, wystarczające do poprawnego zakończenia sesji, zamknięcia zbiorów, zapisania danych itp. Czas ten może również być wykorzystany na podłączenie alternatywnego źródła prądu, np. dużej, znajdującej się w piwnicy baterii akumulatorów, która wystarczy na wiele godzin pracy.

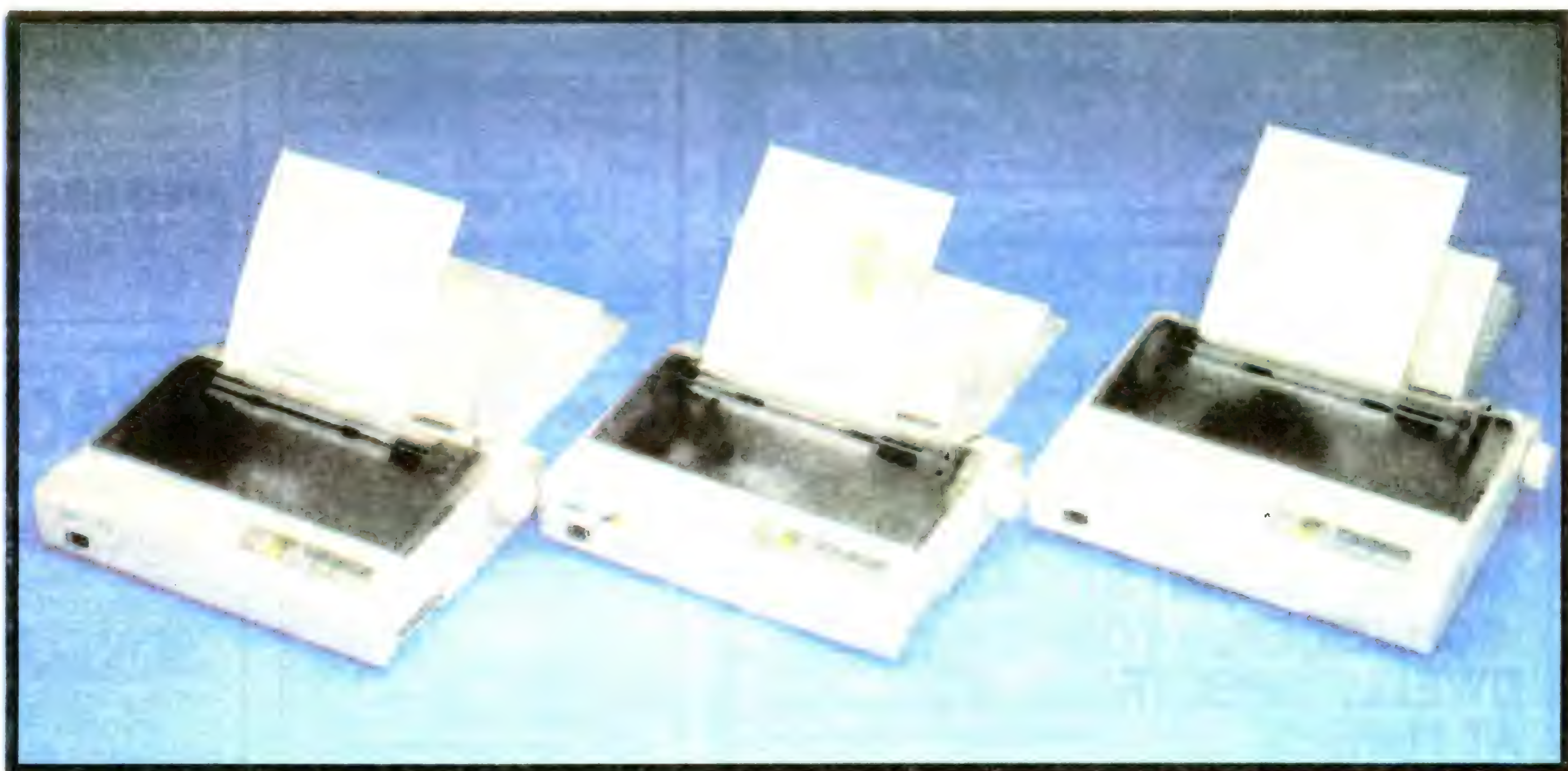
Jeśli nie wiemy jak długo będzie trwała przerwa w dostawie prądu, to energia z akumulatorów, przeznaczona dla komputera może być zbyt cenna na oświetlanie pomieszczeń. Wtedy do dalszej pracy zapalamy lampy naftowe lub świece. W tym ostatnim przypadku należy tylko bardzo uważać aby nie nakapać stearyny na klawiaturę czy dyskietki.

Andrzej Pilaszek

New New New

Drukarka Star

to najlepszy przyjaciel Twojego Komputera!



STAR LC — 10
Najpopularniejsza drukarka świata! Ponad 2.000.000 użytkowników nie może się mylić! Jeśli chcesz mieć drukarkę solidną i niezawodną, wszechstronną i niedrogą, to decyzja jest prosta: LC–10!
Prędkość druku:
144zn/s (draft)
36zn/s (NLQ)

STAR LC — 10 colour
Ta sama japońska jakość i niezawodność jak LC–10, z dodatkową możliwością druku w 7 kolorach!
Wystarczy tylko dzałożyć barwną taśmę.

Prędkość druku:
144zn/s (draft)
36zn/s (NLQ)

STAR LC24 — 10
Drukarka roku 1989 w Europie Zachodniej! Doskonała jakość pisma dzięki nowoczesnej, 24-igłowej głowicy.
Niezastąpiona do korespondencji oraz precyzyjnej grafiki.

Prędkość druku:
170zn/s (draft)
57 zn/s (LQ)

Wszystkie trzy drukarki posiadają cztery kroje czcionek, wbudowany traktor oraz funkcje „Paper Park". Można w nich stosować papier z perforacją lub bez, jak również pojedyńcze kartki.

UWAGA: W celu zainstalowania polskich znaków, prosimy skontaktować się z naszym punktem serwisowym.

Zalecane	LC–10	2 700 000 zł, kaseta barwiąca (czarna) 40 000 zł
ceny	LC–10 ctr	3 900 000 zł, kaseta barwiąca (kolorowa) 100 000 zł
detaliczne	LC24–10	4 500 000 zł, kaseta barwiąca (czarna) 100 000 zł, karta z polskimi znakami 650 000 zł.

Drukarki Star można kupić:

W Warszawie w sklepie Agrocomputer, ul. Grażyny 1, tel. 48-63-94
W Kielcach w firmie Interbit, ul. Man. Lipcowego 4, tel. 441-99
oraz w zasadzie w każdej firmie komputerowej na terenie całego kraju.

Serwis i instalacja znaków polskich:

ABC Data Service, Warszawa, ul. Konopnickiej 6, tel. 28-92-81 w. 128
Interbit, Kielce, ul. Man. Lipcowego 4. tel. 441-99

Przedstawiciel na Polskę:
ABC Data Sp. z o.o.
Warszawa, ul. ZWM 9
tel. 643-53-36 tx. 817123

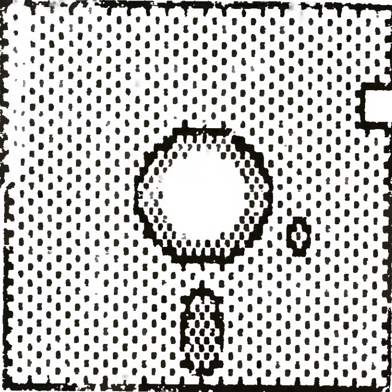
star
the ComputerPrinter

ABC Data

CENY NA DZIEŃ 20.08.1990

	Giełda	Sklep	Pewex	Zachód
	tys. zł		$	
SINCLAIR				
ZX 81	200	—	—	—
ZX Spectrum 48	850	850	—	—
ZX Spectrum +	950	1100	—	—
ZX Spectrum + 2	—	—	—	—
Timex 2048	1000	1100	—	—
stacja FDD3000	850	1500	—	—
stacja FDD3	500	—	—	—
drukarka GP–50	—	450	—	—
Masterface I	100	—	—	—
AY 3–8910	130	—	—	—
COMMODORE				
C 64	1450	1750	199	—
C 64 Desk–Top	2000	—	—	—
C 128	2200	2450	—	—
C 128 D	4300	4700	599	—
Amiga 500	5200	5900	765	—
Amiga 2000	18300	—	—	—
magnetofon	330	350	35	—
stacja 1541	1300	1450	—	—
stacja 1571	1900	2200	—	—
stacja Oceanic	1700	—	—	—
LC 10C	2200	2000	299	—
Final II	90	—	—	—
Final III	200	—	—	—
Action V	300	—	—	—
ATARI				
800 XL	1100	1090	—	—
65 XE	1100	1090	127	—
130 XE	1400	1650	199	—
520 STFM	4200	4400	499	—
1040 STFM	—	—	899	—
magnetofon	340	350	51	—
stacja 1050	1400	1600	—	—
monitor SM124	—	—	199	—
monitor SM1224	—	—	479	—
drukarka 1029	890	1400	—	—
Turbo 2000	100	100	—	—
Centronics	100	75	—	—
AMSTRAD				
464	2000	2300	—	—
664	—	—	—	—
6128	3300	—	—	—
PCW 8256	—	—	—	—
PCW 8512	5100	—	—	—
PC 1512	7000	—	—	—
PC 1640	14.000	—	—	—
IBM				
PC XT stand.	7500	5400	499	—
PC AT stand.	9000	—	—	—
HD 20 MB	2700	2000	370	—
napęd 5″	600	—	100	—
monitor amber	—	—	219	—
klawiatura	750	—	—	—
INNE				
dyskietki 5″	3.3–6	4.5	—	—
dyskietki 3.5″	8–15	8–11	2	—
dyskietki 3″	15–30	29.5	—	—
kasety C–60	14	—	1	—
monitor Neptun	650	—	—	—
joystick	50–90	60–80	7–15	—

Sklepy „Bajtka": Bytom, ul. Koniewa 6, tel. (832) 81-57-01
Kraków, ul. Pstrowskiego 9, tel. (012) 56-54-52



INDYWIDUALNY BANK DANYCH

Cezary Kieloch lat 16 posiada THOMPSON TO7. Nawiąże kontakt w celu wymiany programów i doświadczenia. Proponuje programy edukacyjne i użytkowe. Adres: ul. Powstańców Śl. 14, 42-737 Psary Śl.

Gazda Wojciech lat 17, posiada SHARP MZ-721. Proponuje wymianę firmowych gier oraz programów użytkowych. Adres: 30-014 Kraków, ul. Litewska 28/46.

Michał Siarkiewicz posiada komputer MERITUM 1 mod. 2, nawiąże kontakt z innymi użytkownikami tego typu komputera w celu wymiany programów i literatury. Adres: Kościerzyn 80 m 5, woj. Sieradzkie, 98-291 Charłupia Mała.

Leszek Kieda lat 13, posiada SPECTRAVIDEO 738 w systemie MSX, ze stacją dysków 3,5 cala. Pragnie nawiązać kontakt z użytkownikami tego komputera, oraz firmą która zajmuje się oprogramowaniem w tym systemie. Adres: 71-013 Szczecin, ul. Zielona 2/1

Maciej Andrzejczyk lat 13, posiada ATARI 65XE, monitor, drukarkę oraz stację dysków. Interesuje się programowaniem. Adres: Jadzowskiego 18, 86-300 Grudziądz

Tomasz Durak lat 20, posiada ATARI 520 STM, stację dysków 314. Interesuje się informatyką, elektroniką, grafiką komputerową, muzyką elektroniczną. Pragnie nawiązać kontakt z posiadaczami Atari ST w celu wymiany oprogramowania, literatury i doświadczeń. Adres: Krasickiego 10/34 39-400 Tarnobrzeg.

Bernard Szeryński, student, lat 19 posiada komputer COMMODORE PC 10-III, stację dysków 5,25 oraz 3,5 cala plus dysk twardy 32 MB i drukarkę STAR LC 10. Proponuję wymianę oprogramowania oraz literatury dotyczącej IBM. Adres: 04-853 Warszawa, ul. Borkowska 18

Tomasz Polkowski, uczeń I klasy LO, lat 16. Zainteresowania: modelarstwo, elektronika, języki obce, posiada ATARI XE Video Game System, stacja dysków, California Access 2001, 2 joysticki, magnetofon. Pragnie nawiązać kontakt w celu wymiany gier i programów użytkowych. Adres: 05-310 Kałuszyn, ul. Zamojska 10/28

Daniel Smolarczyk posiada komputer VC 20, pragnie korespondować z kolegami którzy posiadają taki komputer. Adres: ul. Łukasińskiego 58, 43-300 Bielsko-Biała

Mieczysław Szefer lat 42, posiada ZX Spectrum, rozszerzenie do 80 kB RtM, stację dysków 5,25" 720 kB z interface firmy Opus wersja 1.31, modół dźwięku zAY. ma około 1200 gier, 300 programów dydakt. oraz ok. 500 programów użytkowych. Adres: ul. Jagiellońska 16 B/3, 58-560 Jelenia Góra

Paweł Michalski lat 17, posiada COMMODORE C64 magnetofon Datassette Unit, monitor. Nawiąże kontakt w celu wymiany doświadczenia i oprogramowania. Adres: Dobieszyn 252, 38-460 Jedlicze woj. Krosno.

Drogi Bajtku!

Na listy Czytelników odpowiadają autorzy „Bajtka"

Jestem posiadaczem stacji TIMEX FDD 3000 i w związku z tym mam kilka problemów. Mam zamiar zamontować płytę Spectrum w obudowie stacji, i nie wiem, czy zasilacz stacyjny wytrzyma obciążenie dwóch napędów i komputera? (...)
Co zrobić z blokami danych o długości powyżej 41K?

Andrzej Gawlik, Wrocław

Wielu użytkowników stacji FDD 3000 zachęconych zostaje jej rozmiarami do „upychania" różnych interface'ów, a nawet całego Spectrum. Takie rozwiązanie ma swoje dobre strony, ale też i złe. Dobrą stroną może być estetyczniejszy wygląd zestawu, mniej kabli na biurku. Złą stroną bywa w takich przypadkach to, że cały zestaw może działać błędnie, lub w ogóle nie pracować. Problemem będzie na pewno klawiatura, która zamontowana na dodatkowych kablach, bez przetwornika, może nie pracować lub „zawieszać" system.

Dlatego takie przeróbki powinien wykonywać doświadczony elektronik.

Zasilacz w stacji przeznaczony jest do zasilania tylko dwóch napędów. Podłączenie czegoś dodatkowo jest więc ryzykowne.

Bloki danych o długości powyżej 41K należy podzielić. Najlepiej zrobić to za pomocą programu typu COPY-COPY. Tak podzielony blok „zrzucamy" na dysk, a tam już nie mamy problemów z wgraniem tych kilku (bo po podzieleniu mamy kilka) plików.

Przykład: plik na taśmie o długości 49152 bajtów — czyli praktycznie cała pamięć RAM.

Dzielimy go na trzy bloki: pierwszy o długości 6912 (jest to sam obrazek), drugi o długości 1704 (od 23296 do 25000), trzeci o długości 40536 (od 25000 do końca). Tak otrzymane bloki przenosimy na dysk, gdzie pierwszy plik wgrywamy normalnie jak obrazek, trzeci wgrywamy pod adres własny, zaś drugi w ustalone miejsce na obrazku. Z kolei jeszcze w innym miejscu na obrazku posiadamy procedurę maszynową (musimy ją napisać), przenoszącą plik drugi w odpowiednie miejsce. Przykładowo adresem wgrania pliku drugiego początkowo jest 16384, procedurę naszą możemy więc umieścić np. pod adresem 20480, i ma ona wtedy postać:

```
DI
LD SP,22520
LD HL,16384
LD DE,23296
LD BC,1704
LDIR
JP START
```

Stos ustawiamy w pamięci obrazka dlatego, że wyżej nie ma miejsca. START oznacza adres uruchomienia naszego pliku, bo trzeba wiedzieć, że tak długie pliki są z reguły plikami kodu maszynowego gier. Loader w BASIC'u będzie miał postać:

```
10 CLEAR 24999
20 LOAD *"PLIK_1" SCREEN$
30 LOAD *"PLIK_3" CODE 25000
40 CLS : LOAD *"PLIK_2" CODE 16384
50 RANDOMIZE USR 20480
```

Takie postępowanie jest prawidłowe dla większości przypadków, lecz czasami jest potrzebna dokładna analiza oryginalnego loadera oraz samych danych.

(...) Jak komputer sprawdza, czy ma podłączony AY?
Czy można poprawić takie gry, jak „Glider Rider" i „Skate Crazy 2"?

Adam Dawidziuk, Białystok

Komputer rzadko sprawdza AY-greka, raczej rozpoznaje typ komputera, na którym został uruchomiony. Dzieje się to najczęściej na drodze przełączania banków pamięci, lub wyrywkowych testów ROM'u. Procedury zajmujące się rozpoznawaniem typu maszyny są zwykle na samym początku programu, więc należy przeprowadzać wnikliwą analizę działania poszczególnych początkowych procedur. Właściwa procedura używa zwykle rozkazów: LD BC, 32765 oraz OUT (C),A.

Sprawdzanie ROM'u sprowadza się zwykle do testowania napisu producenta, lub jednej z komórek np. 14446. Tu musimy posłużyć się metodą dedukcji.

Jeśli program rozpozna, że został uruchomiony na komputerze z AY-grekiem — czyli na Spectrum 128, to może chcieć wykorzystywać dodatkowe banki pamięci jakimi dysponuje ten model Spectrum. Z takiej przyczyny programy "Skate Crazy 2" albo "Glider Rider" nie mogą działać na Spectrum 48K i dołączonym AY-grekiem.

Czy można podłączyć AY-greka do ELWRO 800 Junior?

Jan Miach, Oświęcim

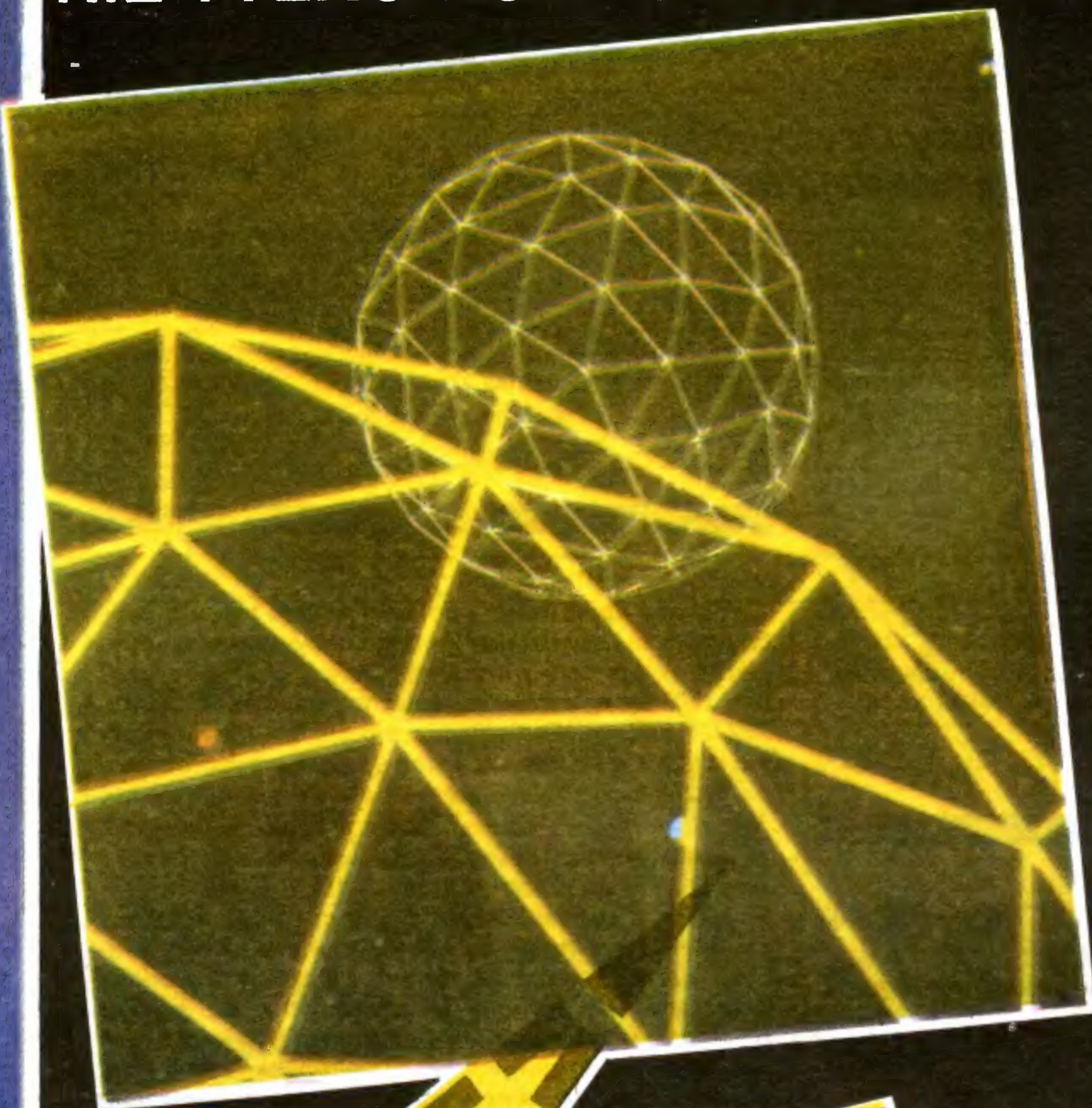
Niestety nic nam o tym nie wiadomo, ponieważ firma ELWRO nie udostępniła nam żadnego egzemplarza tej maszyny, ani też jej dokumentacji...

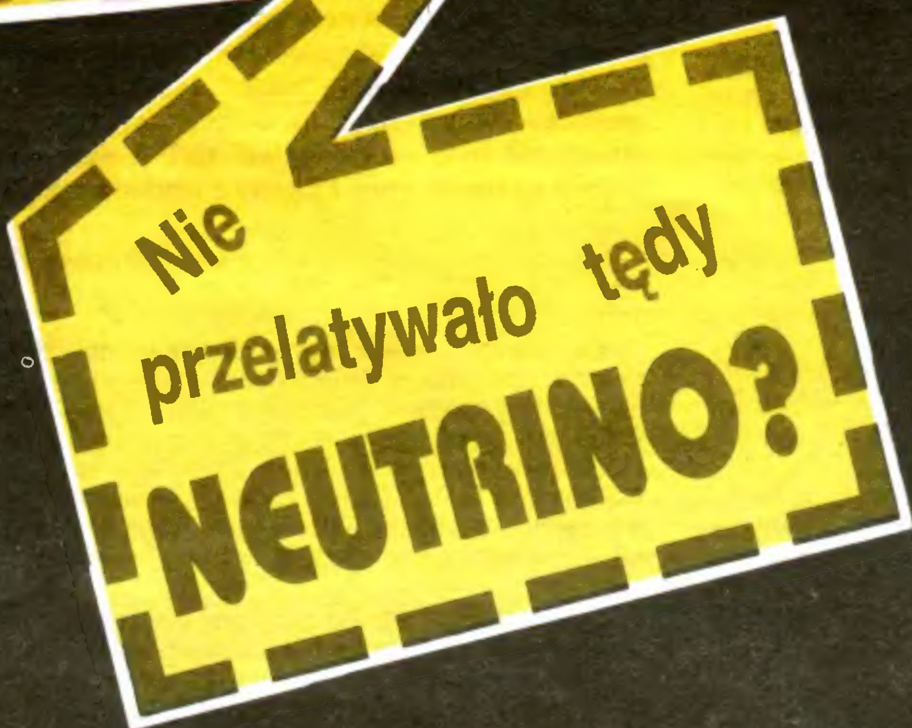
Podczas kopiowania i porządkowania zbiorów próbowałem maksymalnie wykorzystać pojemność dyskietek. W tym celu grupowałem zbiory według rozmiarów tak, by w sumie zajmowały możliwie dokładnie 362496 bajtów. Niestety, okazało się, że prawie nigdy nie udaje się skopiować w ten sposób przygotowanego zestawu na dyskietkę — pojawia się komunikat "Insufficient disk space". Stało się tak na przykład, gdy próbowałem zmieścić na jednej dyskietce trzy zbiory, o rozmiarach 327488, 24072 i 10449 bajtów. Mają one w sumie 362009 bajty, a mimo to nie mieszczą się na normalnej dyskietce (nie ma na niej żadnych uszkodzonych sektorów). Zacząłem dokładniej przyglądać się liczbom pokazywanym przez system podczas różnych operacji, i okazało się, że większość podawanych informacji jest niekonsekwentna, dla przykładu, na tej samej dyskietce o której napisałem przed chwilą, po skopiowaniu największego zbioru (327488 bajty), system "widzi" 34816 bajtów wolnego miejsca. Razem daje to 362304 bajty, czyli o 192 mniej, niż było na początku. Czy z tego wynika, że twórcy systemu mieli kłopoty z dodawaniem?

Piotr Pokorski, Warszawa

Nie, wbrew prezentowanym przez pana liczbom. Wynika to ze sposobu, w jaki zorganizowane są na dysku informacje. Najmniejsza jednostka powierzchni dysku, jaka może być przydzielona zbiorowi, nazywa się klastrem i ma 1024 bajty. Każdy zbiór musi zajmować na dysku wielokrotność tej liczby. Pański zbiór o rozmiarach 327488 bajtów zajmuje na dysku 320 klastrów, czyli 327680 bajty. Teraz suma będzie się zgadzać.

Z rozmiarami klastrów związana jest jeszcze jedna ważna sprawa. O ile w przypadku dużych zbiorów procentowa różnica między rzeczywistymi rozmiarami zbioru a zajmowaną przezeń przestrzenią na dysku jest niewielka, o tyle w przypadku małych zbiorów, liczących kilkadziesiąt lub kilkaset bajtów, procent wykorzystania dysku może być bardzo mały. Istnieje program naszego redakcyjnego kolegi, którym można poprawić wykorzystanie powierzchni dysku, być może napiszemy o nim wkrótce.





Nie przelatywało tędy NEUTRINO?

Neutrino to taka mała cząstka elementarna, która na pewno ma określoną energię, spin równy 1/2 i prędkość zbliżoną do prędkości światła. Być może ma także masę, chociaż tego na razie do końca nie wiadomo. Jednej cechy, dość popularnej wśród innych cząstek, nie ma wcale — nie ma najmniejszej ochoty na jakiekolwiek oddziaływania z materią. Przelecieć na wylot przez Słońce i wszystkie planety na dokładkę — żaden problem, niech się tylko ustawią w odpowiedni sposób.

Neutrina (znamy ich obecnie trzy rodzaje) powstają w wyniku rożnych reakcji jądrowych. Te, ktore najbardziej chcielibyśmy bliżej poznać, są produktami rożnych reakcji syntezy cięższych jąder atomowych z protonow, czyli jąder wodoru. Reakcje te zachodzą wewnątrz Słońca, i są źrodłem energii słonecznej. Ponieważ znamy ilość energii emitowanej przez Słonce, i wydajnośc zachodzących w nim reakcji, możemy obliczyc ilosc powstających neutrin. Jest ona niemal niewyobrażalna — dość powiedzieć, że każdy z nas, leżąc plackiem na plaży, jest w ciągu każdej sekundy przeszywany przez około 10^{15} neutrin. Wszystkie one przelatują między atomami, z ktorych jesteśmy zbudowani, nie zwracając na nie najmniejszej uwagi.

Ta odpornośc neutrin na oddziaływanie z materią spedza sen z oczu fizykom. Powód ich zmartwień jest prosty — badanie cząstek elementarnych polega na obserwowaniu efektów ich oddziaływań z innymi cząstkami, takimi, ktore jesteśmy w stanie zarejestrować. Brak oddziaływan oznacza niemożność badania, a neutrina niosą niezwykle ciekawe informacje — pochodzą przecież z miejsca, do ktorego nigdy nie będziemy mogli zajrzeć — z wnętrza Słońca. Gdyby udało się dokładnie zmierzyć ich ilosc, i pomierzyć ich energie, wiedzielibysmy, czy stworzony przez astronomow teoretyczny model Słońca, (tak zwany model standardowy) opisujący jego budowę i ewolucję, jest dobry, czy nie.

To, że neutrina bardzo niechętnie oddziaływują z materią, na szczęscie nie znaczy, że nie robią tego wcale. W 1968 roku zaprojektowano i wykonano pierwszy detektor neutrin. Zasada jego działania opiera się na reakcji, w ktorej neutrino (o odpowiednio wysokiej energii) w zderzeniu z atomem chloru 37 (trwały izotop, stanowiący 25% całego chloru na ziemi) zmienia go w atom argonu 37 — radioaktywnego izotopu. Detektor jest ukrytym w starej kopalni głęboko pod ziemią olbrzymim zbiornikiem o pojemności około 400 tysięcy litrow, wypełnionym sześciochloroetanem (używanym czasem jako rozpuszczalnik do prania chemicznego). Raz na kilka miesięcy cały rozpuszczalnik jest przeglądany cząsteczka za cząsteczką w celu wydobycia wszystkich atomow argonu 37. Ich ilośc odpowiada ilości reakcji neutrin z chlorem 37. Rownolegle z przygotowaniem detektora obliczono, ile neutrin powinien on wykrywać. Trzeba było w tym celu zdefiniować nową jednostkę — nazwano ją SNU (solar-neutrino unit), a odpowiada ona jednemu przejściu atomu chloru 37 w argon 37 pod wpływem neutrina na sekundę na 10^{36} atomów chloru (to pokazuje, jak mało reaktywne są neutrina). Z obliczen wynikało, że obserwowany strumien neutrin powinien wynosić około 7.9 ± 2.6 SNU. Tymczasem pomiary wykazały deficyt neutrin. Wynik wynosił 2.7 SNU. Początkowo szukano błędu w eksperymencie (a o skali trudności niech świadczy jedna liczba — dziennie powstaje w zbiorniku średnio 0.46 atomu argonu 37) i obliczeniach, ale po dwudziestu latach, ktore upłynęły od tamtego czasu, nie ma wątpliwości — obliczenia i eksperyment są w porządku.

Z obliczeniami sprawa nie jest może tak prosta, bo do ich wykonywania posłużono się standardowym modelem Słonca, ktory może okazać się błędny. Jest to jednak bardzo mało prawdopodobne. Za poprawnością modelu przemawia bardzo wiele faktów — przede wszystkim jest on w bardzo dobrej zgodności ze wszystkimi obserwacjami Słońca (oczywiście oprocz obserwacji neutrin), do tego stopnia, że przewidziane na jego podstawie częstotliwości fal akustycznych na powierzchni Słonca (a przewidziano je przed opracowaniem technik pomiarowych) są zgodne z wynikami pomiarów w granicach jednego procenta. Nie ma drugiego teoretycznego modelu Słońca, ktory można by było w podobny sposób zweryfikować przy pomocy obserwacji. Swoją drogą, model ten jest powszechnie używany do interpretacji wynikow obserwacji innych gwiazd, i jego zmienienie postawiłoby na głowie dużą częśc (jeśli nie większość) prac astronomów zajmujących się ewolucją gwiazd.

W 1986 w Kamioka, w Japonii rozpoczęto eksperyment o nazwie Kamiokande II. Badaniom poddano neutrina o znacznie większych energiach, tak dużych, że mogą wybić elektrony z cząstek wody. Wynik — 3.2 SNU, czyli nieco więcej, ale i tak za mało. Obserwacja wybitych elektronów pozwala nie tylko na policzenie neutrin i określenie ich energii, ale także na określenie kierunku, z ktorego przyleciały. Szczęśliwym zbiegiem okoliczności eksperyment został przygotowany na kilka miesięcy przed wybuchem supernowej w lutym 1987 roku. Kamiokande II zarejestrował strumien neutrin pochodzących od tej supernowej, i — o ironio — potwierdził teorie o przebiegu wybuchu i zachodzacych w nim zjawiskach fizycznych. Coż, z teorią supernowych udało nam się lepiej niż z własnym Słoncem.

Deficyt neutrin od lat nie daje spokoju tak teoretykom jak i eksperymentatorom. Ci pierwsi maja w zanadrzu kilka teorii, ktore mogą wytłumaczyc obserwowane braki. Najciekawsza z nich znana jest jako efekt Michiejewa-Smirnowa-Wolfensteina (MSW). MSW mowi, że jeśli masa ktoregokolwiek z rodzajów neutrin jest większa od zera (a wyznaczenie masy neutrina też się jeszcze nikomu nie udało...), możliwe są przejscia neutrin elektronowych (stosunkowo łatwo oddziaływujących z materią) w neutrina mionowe, znacznie trudniejsze do obserwacji. Żeby móc ocenic, czy tak jest w rzeczywistości, trzeba by zbadać dokładnie neutrina o najniższych energiach, te ktore nie są obserwowane w detektorze z chlorem 37.

Co na to eksperymentatorzy? Neutrina o niskich energiach mogą powodować zmianę galu 71 (stabilny) w german 71 (radioaktywny). Postanowiono to wykorzystać w konstrukcji następnych detektorow. Przygotowano dwa eksperymenty — w Połnocnym Kaukazie radziecko-amerykański SAGE (Soviet-American Gallium Experiment) i europejsko-amerykański GALLEX we Włoszech. SAGE — to ukryte w kopalni 60 ton metalicznego galu, GALLEX — to trzydzieści ton jego chlorku. O skali obu eksperymentów niech świadczy jeden fakt — w trakcie ich projektowania roczna światowa produkcja galu wynosiła dziesięć ton. Na wyniki trzeba jeszcze trochę poczekać, wstępne rezultaty SAGE mają być ogłoszone w tym roku.

Inne wytłumaczenie proponowane przez teoretyków mówi, że ilośc neutrin emitowanych przez Słońce zmienia się w sposób cykliczny, raz jest większa, raz mniejsza. Wprawdzie możliwośc taka jest wykluczona przez model standardowy, ale według niej deficyt obserwowanych neutrin też nie powinien mieć miejsca.

Eksperymentatorzy znaleźli już sposób na potwierdzenie (lub obalenie) i tej koncepcji. Jeśli podda się naświetlaniu promieniowaniem dowolną probkę, i będą w niej powstawać radioaktywne pierwiastki, po jakimś czasie ustali się rownowaga — w jednostce czasu tyle samo atomów radioaktywnych powstaje ile się rozpada, i ich ilośc pozostaje dzięki temu stała. Jeśli szybkości powstawania i rozpadania się atomów są małe, układ będzie w stanie rownowagi zależnym od średniego natężenia promieniowania, a nie od jego chwilowych wahań. Molibden, poddany działaniu neutrin o bardzo wysokich energiach zmienia się w nietrwały technet, o czasie połtrwania rownym 4.2 miliona lat. Ta para pierwiastków spełnia opisane warunki, i być może dokładne zbadanie ilości technetu w rudach molibdenu pozwoli na wyznaczenie średniego strumienia neutrin na przestrzeni ostatnich kilku milionów lat. Być może — bo jest wiele czynników, ktore trzeba wziąć pod uwagę, i nie wiadomo, czy uda się je wszystkie dokładnie zbadać.

Projektowane są rownież inne detektory — jeden z nich, składający się przede wszystkim ze zbiornika z milionem litrow ciężkiej wody (czyli takiej, w ktorej zamiast wodoru jest deuter), ma pozwolić na obserwację rozpraszania i absorpcji neutrin, ale tylko tych o najwyższych energiach. Ciekawie zapowiadają się urządzenia oparte na bardzo niskich temperaturach — w tak niskich temperaturach bardzo małe energie oddziaływań mogą dawać wielkie, łatwo zauważalne efekty. To wszystko jednak przyszłość, która może, ale nie musi, przynieść odpowiedzi na pytanie o brakujące 5 SNU.

Przepraszam, nie przelatywało tędy neutrino?

Opracował (na podstawie Scientific American)
Marcin Borkowski